In [1]:

```
import os
```

In [2]:

```
import pandas as pd
from pandas import DataFrame
import numpy as np
```

In [3]:

```
import urllib
from urllib.request import urlretrieve
```

In [4]:

```
url="https://github.com/swapniljariwala/nsepy"
```

In [5]:

```
from nsepy import get_history
from datetime import date
```

In [6]:

```
nse_tcs=get_history(symbol="TCS",start=date(2015,1,1),end=date(2015,12,31))
```

In [7]:

```
nse_tcs.columns
```

Out[7]:

```
Index(['Symbol', 'Series', 'Prev Close', 'Open', 'High', 'Low', 'Last',
       'Close', 'VWAP', 'Volume', 'Turnover', 'Trades', 'Deliverable Volume',
       '%Deliverble'],
      dtype='object')
```

In [8]:

```
nse_infy=get_history(symbol="INFY",start=date(2015,1,1),end=date(2015,12,31))
```

In [9]:

```
nse_infy.columns
```

Out[9]:

```
Index(['Symbol', 'Series', 'Prev Close', 'Open', 'High', 'Low', 'Last',
       'Close', 'VWAP', 'Volume', 'Turnover', 'Trades', 'Deliverable Volume',
       '%Deliverble'],
      dtype='object')
```

In [10]:

```
nifty_it=get_history(symbol="NIFTY IT",start=date(2015,1,1),end=date(2015,12,31))
```

In [11]:

```
nifty_it.columns
```

Out[11]:

```
Index(['Symbol', 'Series', 'Prev Close', 'Open', 'High', 'Low', 'Last',
       'Close', 'VWAP', 'Volume', 'Turnover', 'Trades', 'Deliverable Volume',
       '%Deliverble'],
      dtype='object')
```

In [12]:

```
nse_infy.head()
```

Out[12]:

	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	
Date											
2015-01-01	INFY	EQ	1972.55	1968.95	1982.00	1956.9	1971.00	1974.40	1971.34	500691	9.8
2015-01-02	INFY	EQ	1974.40	1972.00	2019.05	1972.0	2017.95	2013.20	2003.25	1694580	3.3
2015-01-05	INFY	EQ	2013.20	2009.90	2030.00	1977.5	1996.00	1995.90	2004.59	2484256	4.9
2015-01-06	INFY	EQ	1995.90	1980.00	1985.00	1934.1	1965.10	1954.20	1954.82	2416829	4.7
2015-01-07	INFY	EQ	1954.20	1965.00	1974.75	1950.0	1966.05	1963.55	1962.59	1812479	3.5

In [13]:

Out[13]:

	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	
Date											
2015-01-01	TCS	EQ	2558.25	2567.0	2567.00	2541.00	2550.00	2545.55	2548.51	183415	4.6
2015-01-02	TCS	EQ	2545.55	2551.0	2590.95	2550.60	2588.40	2579.45	2568.19	462870	1.1
2015-01-05	TCS	EQ	2579.45	2581.0	2599.90	2524.65	2538.10	2540.25	2563.94	877121	2.2
2015-01-06	TCS	EQ	2540.25	2529.1	2529.10	2440.00	2450.05	2446.60	2466.90	1211892	2.9
2015-01-07	TCS	EQ	2446.60	2470.0	2479.15	2407.45	2426.90	2417.70	2433.96	1318166	3.2

In [14]:

```
nifty_it.head()
```

Out[14]:

	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	D
Date													

In [15]:

```
nse_infy.shape
```

Out[15]:

```
(248, 14)
```

In [16]:

```
nse_tcs.shape
```

Out[16]:

```
(248, 14)
```

In [17]:

Out[17]:

	Prev Close	Open	High	Low	Last	Close	VW
count	248.000000	248.000000	248.000000	248.000000	248.000000	248.000000	248.000(
mean	1551.474798	1550.506855	1566.266532	1530.085887	1548.084879	1547.978226	1548.13
std	529.396894	530.578342	534.714088	524.194873	529.493276	529.468189	528.861
min	937.500000	941.000000	952.100000	932.650000	935.500000	937.500000	941.180(
25%	1085.912500	1088.000000	1099.975000	1067.150000	1086.875000	1085.912500	1085.90
50%	1149.650000	1150.000000	1159.725000	1131.150000	1145.625000	1149.325000	1146.24
75%	2125.312500	2136.137500	2150.000000	2104.500000	2125.250000	2125.312500	2125.08
max	2324.700000	2328.500000	2336.000000	2292.050000	2323.200000	2324.700000	2322.17(

In [18]:

```
nse_tcs.describe()
```

Out[18]:

	Prev Close	Open	High	Low	Last	Close	VW
count	248.000000	248.000000	248.000000	248.000000	248.000000	248.000000	248.000(
mean	2538.207460	2542.172782	2563.580444	2514.408468	2538.039718	2537.717944	2538.43
std	86.829359	87.605699	90.598368	82.952778	86.849305	87.057814	86.8130
min	2319.800000	2319.400000	2343.900000	2315.250000	2321.000000	2319.800000	2322.27(
25%	2495.312500	2499.500000	2518.900000	2472.100000	2497.500000	2495.150000	2496.66
50%	2543.050000	2548.500000	2566.000000	2520.000000	2540.150000	2541.475000	2540.44
75%	2592.000000	2594.250000	2615.750000	2567.300000	2593.425000	2592.000000	2592.60
max	2776.000000	2788.000000	2812.100000	2721.900000	2785.100000	2776.000000	2763.04(

In [19]:

Out[19]:

```
Symbol                0
Series                0
Prev Close            0
Open                  0
High                  0
Low                   0
Last                  0
Close                 0
VWAP                  0
Volume                0
Turnover              0
Trades                0
Deliverable Volume    0
%Deliverble           0
dtype: int64
```

In [20]:

```
nse_tcs.isnull().sum()
```

Out[20]:

```
Symbol                0
Series                0
Prev Close            0
Open                  0
High                  0
Low                   0
Last                  0
Close                 0
VWAP                  0
Volume                0
Turnover              0
Trades                0
Deliverable Volume    0
%Deliverble           0
dtype: int64
```

In [21]:

```
<class 'pandas.core.frame.DataFrame'>
Index: 248 entries, 2015-01-01 to 2015-12-31
Data columns (total 14 columns):
Symbol                248 non-null object
Series                248 non-null object
Prev Close            248 non-null float64
Open                  248 non-null float64
High                  248 non-null float64
Low                   248 non-null float64
Last                  248 non-null float64
Close                 248 non-null float64
VWAP                  248 non-null float64
Volume                248 non-null int64
Turnover              248 non-null float64
Trades                248 non-null int64
Deliverable Volume    248 non-null int64
%Deliverble           248 non-null float64
dtypes: float64(9), int64(3), object(2)
memory usage: 29.1+ KB
```

In [22]:

```
nse_tcs.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 248 entries, 2015-01-01 to 2015-12-31
Data columns (total 14 columns):
Symbol                248 non-null object
Series                248 non-null object
Prev Close            248 non-null float64
Open                  248 non-null float64
High                  248 non-null float64
Low                   248 non-null float64
Last                  248 non-null float64
Close                 248 non-null float64
VWAP                  248 non-null float64
Volume                248 non-null int64
Turnover              248 non-null float64
Trades                248 non-null int64
Deliverable Volume    248 non-null int64
%Deliverble           248 non-null float64
dtypes: float64(9), int64(3), object(2)
memory usage: 29.1+ KB
```

In [23]:

```
nifty_it.shape
```

Out[23]:

```
(0, 14)
```

In [24]:

```
nse_infy['Series'].unique()
```

Out[24]:

```
array(['EQ'], dtype=object)
```

In [25]:

```
nse_tcs['Series'].unique()
```

Out[25]:

```
array(['EQ'], dtype=object)
```

In [26]:

```
def movingaverage(x,w):
   return pd.Series(x.rolling(window=w,min_periods=0).mean())
```

In [27]:

```
nse_infy['4weeks']=movingaverage(nse_infy['Close'],20)
nse_infy['16weeks']=movingaverage(nse_infy['Close'],80)
nse_infy['28weeks']=movingaverage(nse_infy['Close'],140)
nse_infy['40weeks']=movingaverage(nse_infy['Close'],200)
nse_infy['52weeks']=movingaverage(nse_infy['Close'],260)
nse_tcs['4weeks']=movingaverage(nse_tcs['Close'],20)
nse_tcs['16weeks']=movingaverage(nse_tcs['Close'],80)
nse_tcs['28weeks']=movingaverage(nse_tcs['Close'],140)
nse_tcs['40weeks']=movingaverage(nse_tcs['Close'],200)
nse_tcs['52weeks']=movingaverage(nse_tcs['Close'],260)
```

In [28]:

```
nse_infy[['Close','4weeks','16weeks','28weeks','40weeks','52weeks']].tail()
```

Out[28]:

	Close	4weeks	16weeks	28weeks	40weeks	52weeks
Date						
2015-12-24	1096.35	1071.865	1099.681250	1124.790357	1418.08950	1555.317418
2015-12-28	1104.75	1073.785	1099.955625	1118.195357	1412.37525	1553.478367
2015-12-29	1104.70	1074.625	1099.876250	1111.717143	1406.94850	1551.654065
2015-12-30	1086.30	1074.990	1099.775000	1105.190357	1401.39000	1549.770040
2015-12-31	1105.40	1077.225	1099.980000	1098.856786	1396.00600	1547.978226

In [29]:

```
nse_tcs[['Close','4weeks','16weeks','28weeks','40weeks','52weeks']].tail()
```

Out[29]:

	Close	4weeks	16weeks	28weeks	40weeks	52weeks
Date						
2015-12-24	2434.25	2382.1025	2489.293750	2523.807500	2534.58675	2539.263934
2015-12-28	2462.70	2387.5700	2487.894375	2522.786429	2533.41800	2538.951429
2015-12-29	2455.80	2392.1250	2486.456250	2521.928929	2532.46300	2538.613415
2015-12-30	2418.30	2394.9450	2484.634375	2520.792143	2531.34100	2538.126316
2015-12-31	2436.85	2398.5275	2483.377500	2519.900714	2530.49250	2537.717944

In [30]:

```
nse_infy.tail()
```

Out[30]:

	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	
Date											
2015-12-24	INFY	EQ	1100.85	1102.05	1104.45	1090.1	1095.8	1096.35	1095.84	615027	6.73
2015-12-28	INFY	EQ	1096.35	1090.00	1110.00	1090.0	1103.8	1104.75	1105.51	2763476	3.05
2015-12-29	INFY	EQ	1104.75	1101.25	1110.00	1097.3	1103.0	1104.70	1104.14	1672531	1.84
2015-12-30	INFY	EQ	1104.70	1101.10	1106.60	1083.0	1088.0	1086.30	1094.45	2576985	2.82
2015-12-31	INFY	EQ	1086.30	1090.15	1109.95	1087.0	1107.0	1105.40	1102.09	3971969	4.37

In [31]:

Out[31]:

	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	
Date											
2015-12-24	TCS	EQ	2425.80	2421.00	2438.2	2404.00	2435.00	2434.25	2424.03	421580	1.0
2015-12-28	TCS	EQ	2434.25	2428.00	2466.4	2420.20	2456.95	2462.70	2455.00	1852099	4.5
2015-12-29	TCS	EQ	2462.70	2458.35	2465.3	2445.75	2449.35	2455.80	2452.26	854262	2.0
2015-12-30	TCS	EQ	2455.80	2453.05	2459.0	2412.30	2421.70	2418.30	2442.06	802881	1.9
2015-12-31	TCS	EQ	2418.30	2415.75	2448.5	2407.50	2430.00	2436.85	2428.90	620159	1.5

In [32]:

```
def volumeshocks(data):
    data['PreviousVolume']=data['Volume'].shift(1)
    data['VolumeShocks'] = (data['Volume']>data['PreviousVolume']*0.1+data['PreviousVolum
e']).map({True:0,False:1})
    return data
```

In [33]:

```
nse_infy=volumeshocks(nse_infy)
nse_tcs=volumeshocks(nse_tcs)
```

In [34]:

```
nse_infy[['Volume','PreviousVolume','VolumeShocks']].head()
```

Out[34]:

	Volume	PreviousVolume	VolumeShocks
Date			
2015-01-01	500691	NaN	1
2015-01-02	1694580	500691.0	0
2015-01-05	2484256	1694580.0	0
2015-01-06	2416829	2484256.0	1
2015-01-07	1812479	2416829.0	1

In [35]:

```
nse_tcs[['Volume','PreviousVolume','VolumeShocks']].head()
```

Out[35]:

	Volume	PreviousVolume	VolumeShocks
Date			
2015-01-01	183415	NaN	1
2015-01-02	462870	183415.0	0
2015-01-05	877121	462870.0	0
2015-01-06	1211892	877121.0	0
2015-01-07	1318166	1211892.0	1

In [36]:

```
def priceshocks(data):
    data['T']=data['Close'].shift(1)
    data['PriceShocks'] = (data['Close']-data['T']>0.20*(data['Close']-data['T'])).map({True:0,False:1})
    return data
```

In [37]:

```
nse_infy=priceshocks(nse_infy)
nse_tcs=priceshocks(nse_tcs)
```

In [38]:

```
nse_infy[['Close','PriceShocks']].head()
```

Out[38]:

	Close	PriceShocks
Date		
2015-01-01	1974.40	1
2015-01-02	2013.20	0
2015-01-05	1995.90	1
2015-01-06	1954.20	1
2015-01-07	1963.55	0

In [39]:

```
nse_infy[['Close','PriceShocks']].head()
```

Out[39]:

	Close	PriceShocks
Date		
2015-01-01	1974.40	1
2015-01-02	2013.20	0
2015-01-05	1995.90	1
2015-01-06	1954.20	1
2015-01-07	1963.55	0

In [40]:

```
def blackswan(data):
    data['T1']=data['Prev Close'].shift(1)
    data['BlackSwanPrice'] = (data['Prev Close']-data['T1']>0.20*(data['Prev Close']-data
['T1'])).map({True:0,False:1})
    return data
```

In [41]:

```
nse_infy=blackswan(nse_infy)
nse_tcs=blackswan(nse_tcs)
```

In [42]:

```
nse_infy[['Prev Close','BlackSwanPrice']].head()
```

Out[42]:

	Prev Close	BlackSwanPrice
Date		
2015-01-01	1972.55	1
2015-01-02	1974.40	0
2015-01-05	2013.20	0
2015-01-06	1995.90	1
2015-01-07	1954.20	1

In [43]:

```
nse_tcs[['Prev Close','BlackSwanPrice']].head()
```

Out[43]:

	Prev Close	BlackSwanPrice
Date		
2015-01-01	2558.25	1
2015-01-02	2545.55	1
2015-01-05	2579.45	0
2015-01-06	2540.25	1
2015-01-07	2446.60	1

In [44]:

```
def priceshocknovolshock(data):
    data['notvolshock']  = (~(data['VolumeShocks'].astype(bool))).astype(int)
    data['PriceshockNovolumeshocks'] = data['notvolshock'] & data['PriceShocks']
    return data
```

In [45]:

```
nse_infy=priceshocknovolshock(nse_infy)
nse_tcs=priceshocknovolshock(nse_tcs)
```

In [46]:

```
nse_infy[['VolumeShocks','PriceShocks','PriceshockNovolumeshocks']].head()
```

Out[46]:

	VolumeShocks	PriceShocks	PriceshockNovolumeshocks
Date			
2015-01-01	1	1	0
2015-01-02	0	0	0
2015-01-05	0	1	1
2015-01-06	1	1	0
2015-01-07	1	0	0

In [47]:

Out[47]:

	VolumeShocks	PriceShocks	PriceshockNovolumeshocks
Date			
2015-01-01	1	1	0
2015-01-02	0	0	0
2015-01-05	0	1	1
2015-01-06	0	1	1
2015-01-07	1	1	0

In [48]:

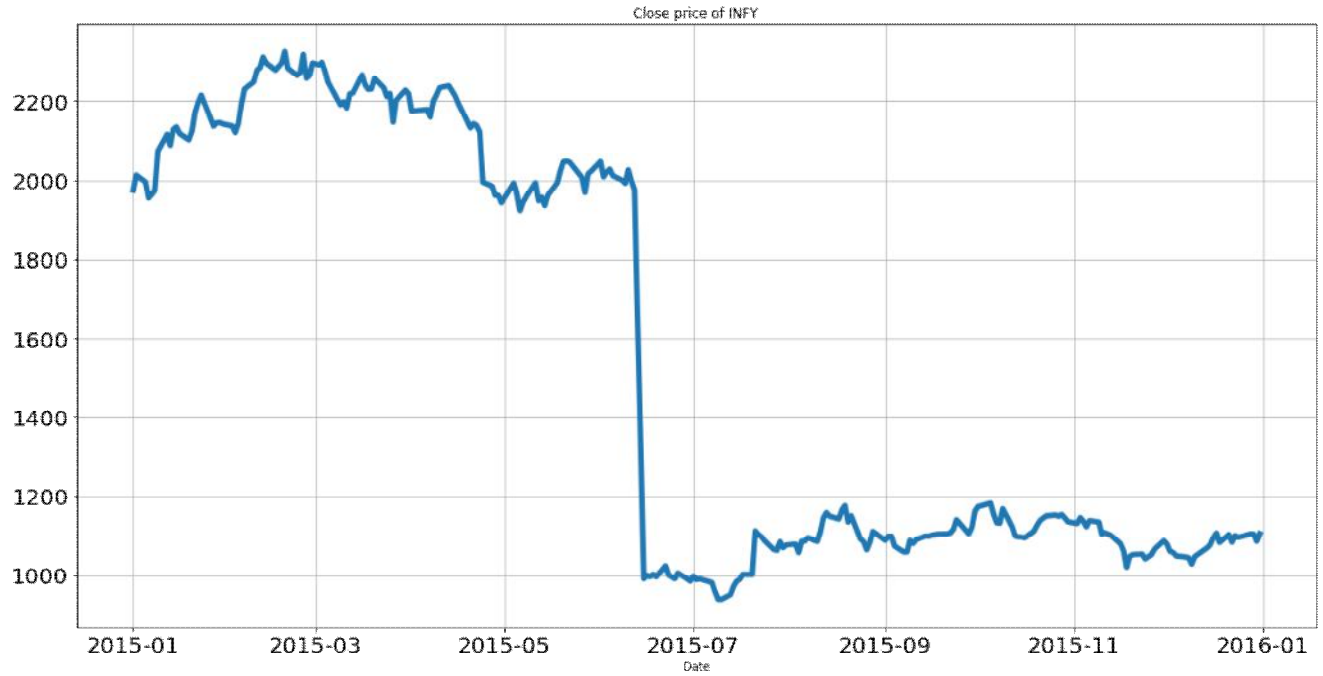
```
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [49]:

```
nse_infy.Close.plot(figsize=(20,10),linewidth=5,fontsize=20,grid=True)
plt.title("Close price of INFY")
```

Out[49]:

```
Text(0.5,1,'Close price of INFY')
```

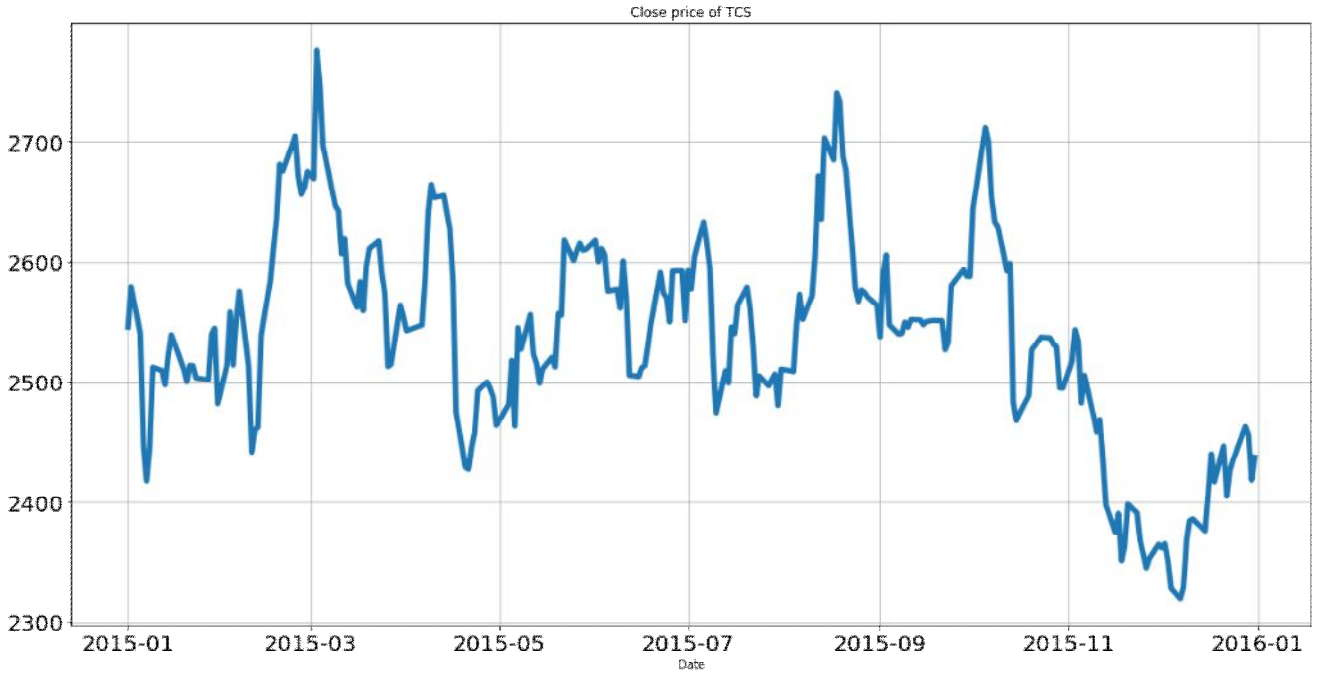


In [50]:

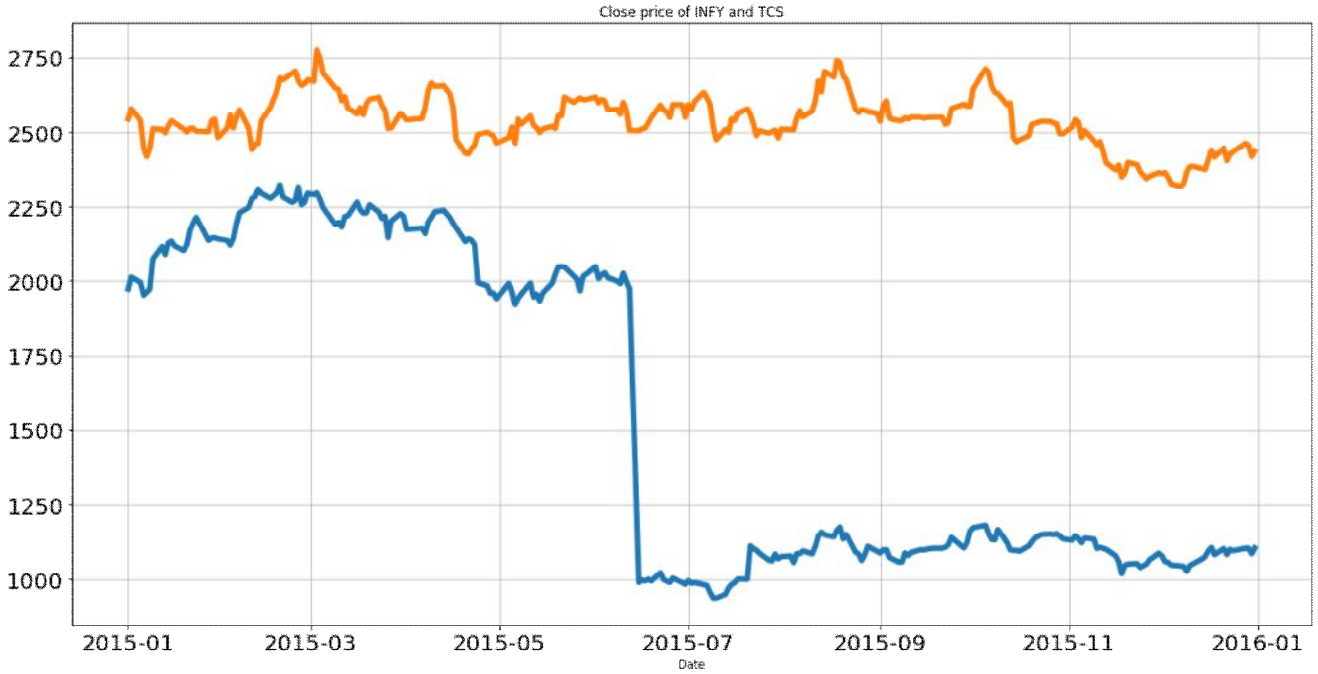
```
nse_tcs.Close.plot(figsize=(20,10),linewidth=5,fontsize=20,grid=True)
plt.title("Close price of TCS")
```

Out[50]:

```
Text(0.5,1,'Close price of TCS')
```



In [51]:

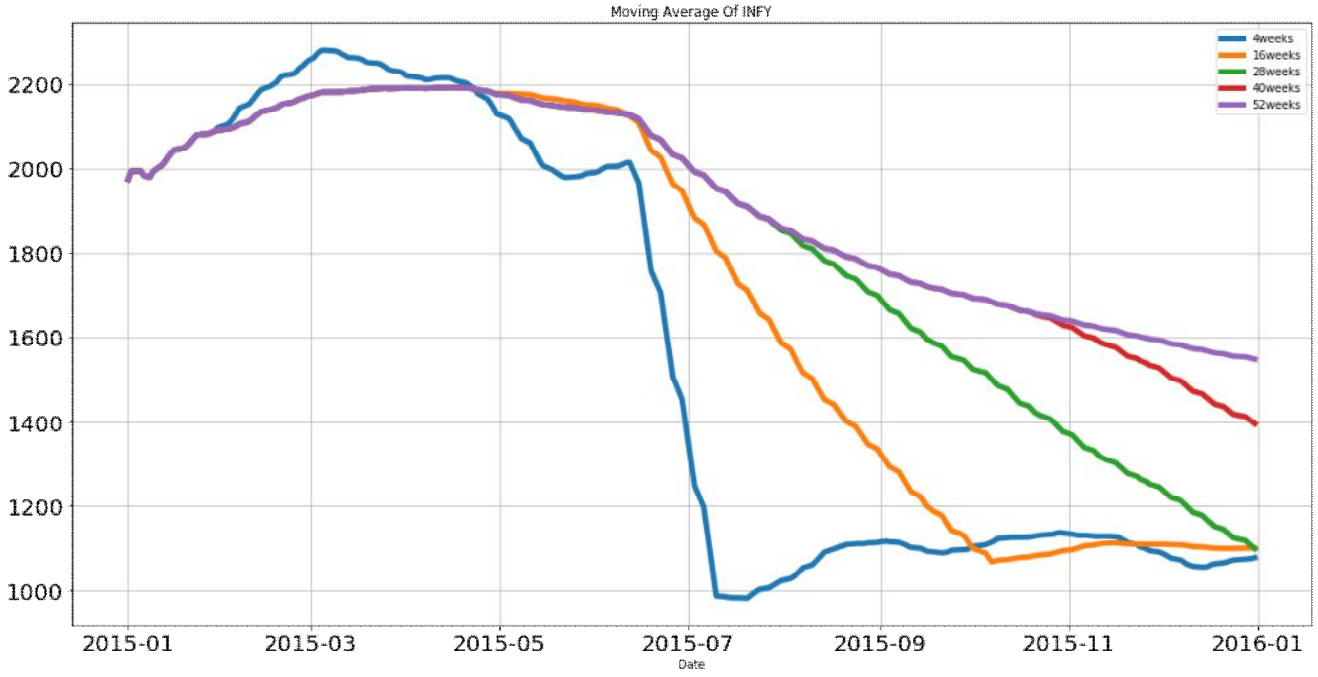


In [52]:

```
nse_infy[['4weeks','16weeks','28weeks','40weeks','52weeks']].plot(grid=True,figsize=(20,1
0),linewidth=5,fontsize=20)
plt.title("Moving Average Of INFY")
```

Out[52]:

```
Text(0.5,1,'Moving Average Of INFY')
```

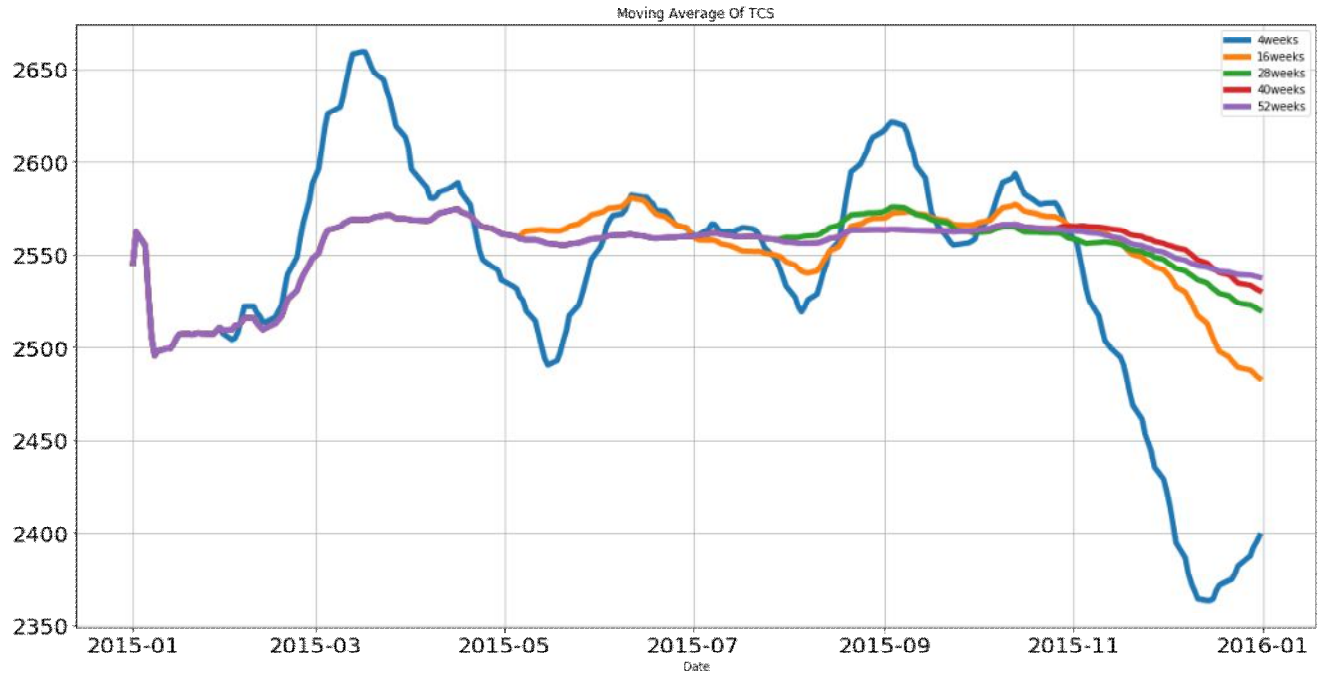


In [53]:

```
nse_tcs[['4weeks','16weeks','28weeks','40weeks','52weeks']].plot(grid=True,figsize=(20,10),
linewidth=5,fontsize=20)
plt.title("Moving Average Of TCS")
```

Out[53]:

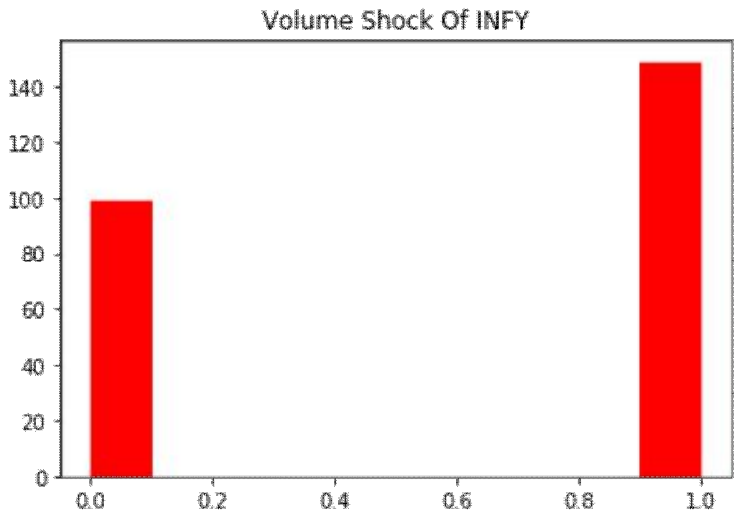
```
Text(0.5,1,'Moving Average Of TCS')
```



In [54]:

Out[54]:

```
Text(0.5,1,'Volume Shock Of INFY')
```

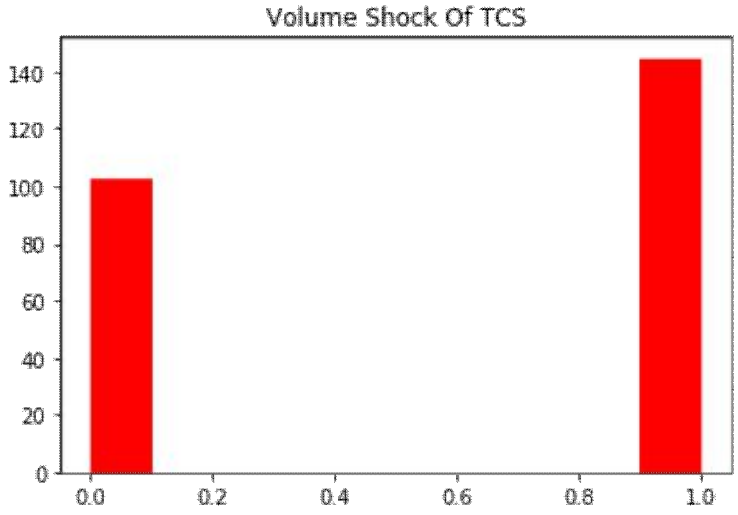


In [55]:

```
plt.hist(nse_tcs.VolumeShocks,color='red')
plt.title("Volume Shock Of TCS")
```

Out[55]:

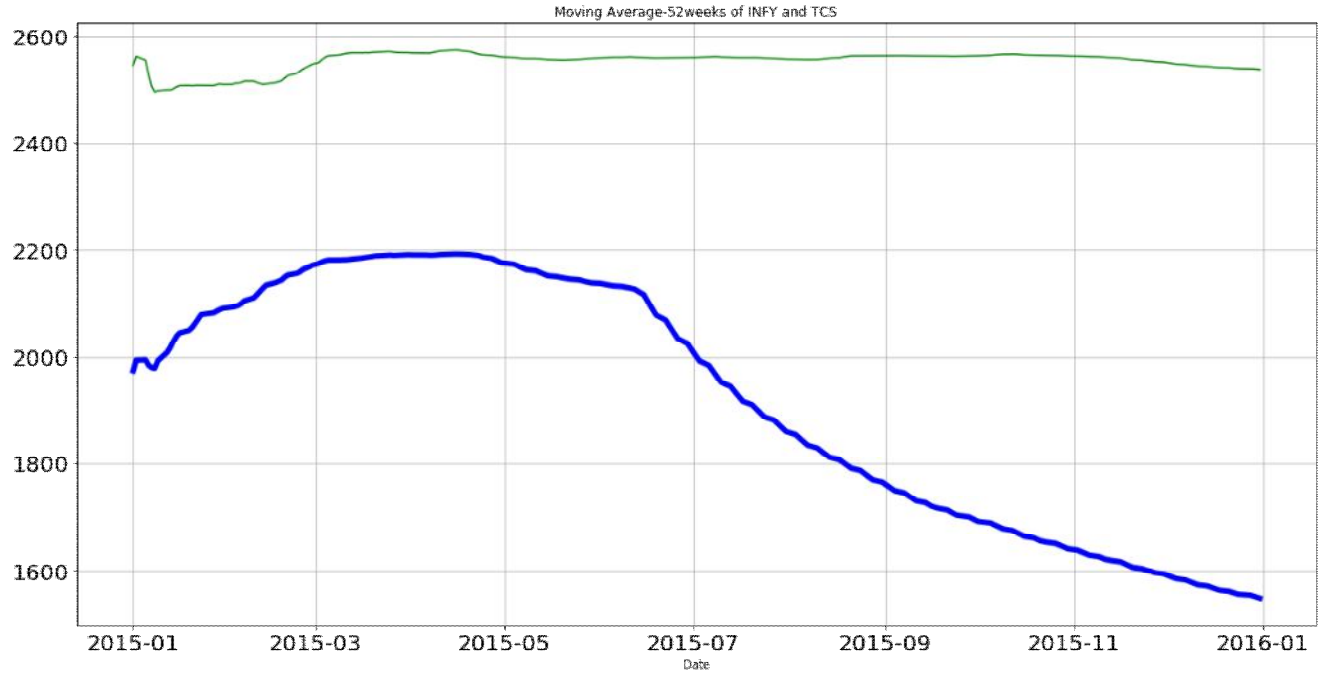
```
Text(0.5,1,'Volume Shock Of TCS')
```



In [56]:

Out[56]:

```
Text(0.5,1,'Moving Average-52weeks of INFY and TCS')
```

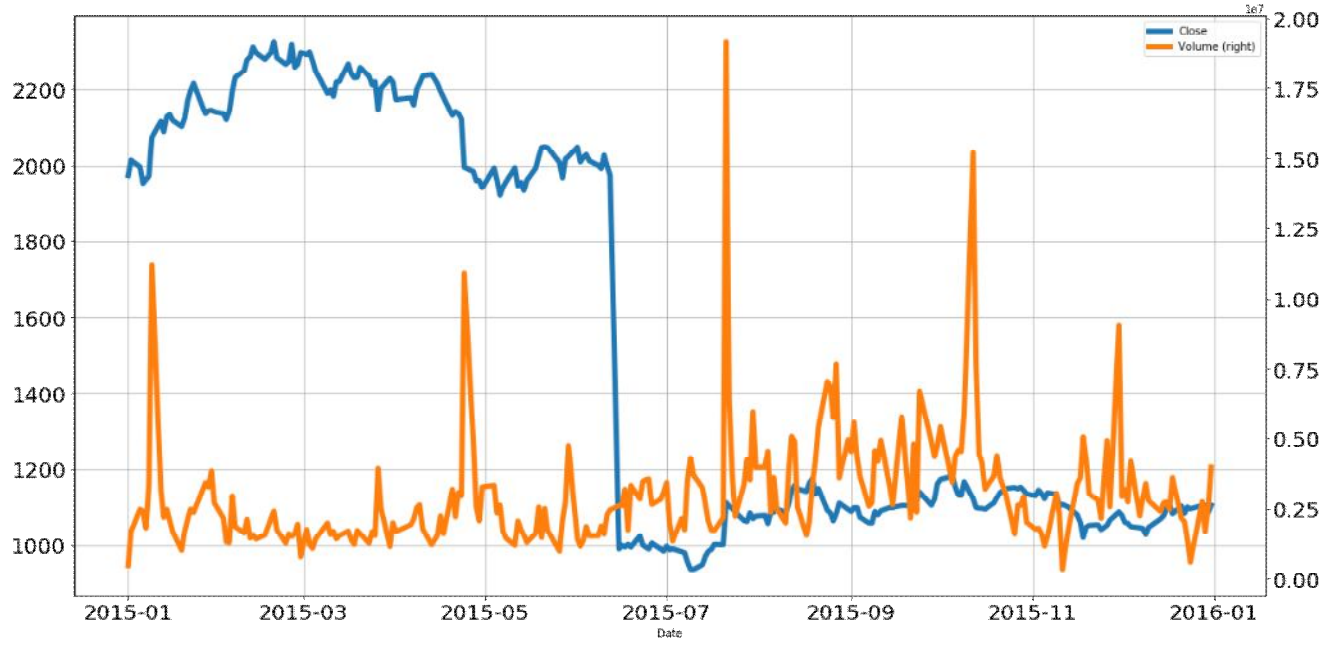


In [57]:

```
nse_infy[['Close','Volume']].plot(secondary_y=['Volume'],grid=True,figsize=(20,10),linewidth=5,fontsize=20)
```

Out[57]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x204f2472e10>
```

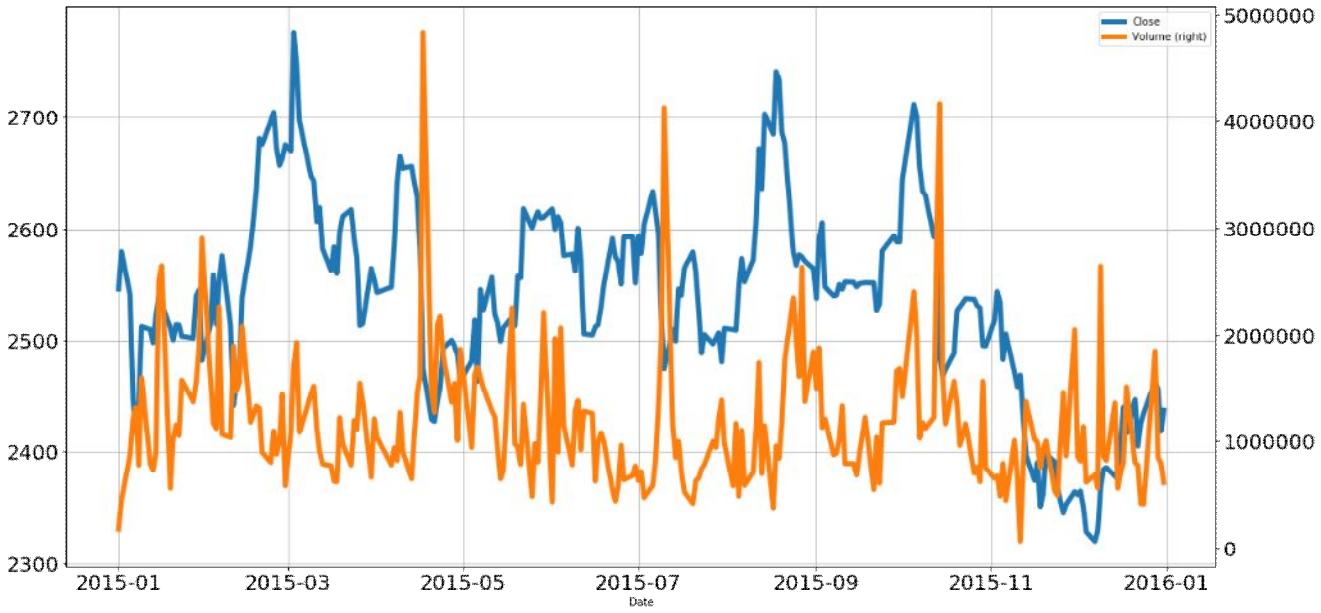


In [58]:

```
nse_tcs[['Close','Volume']].plot(secondary_y=['Volume'],grid=True,figsize=(20,10),linewidth
=5,fontsize=20)
```

Out[58]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x204f3057e80>
```

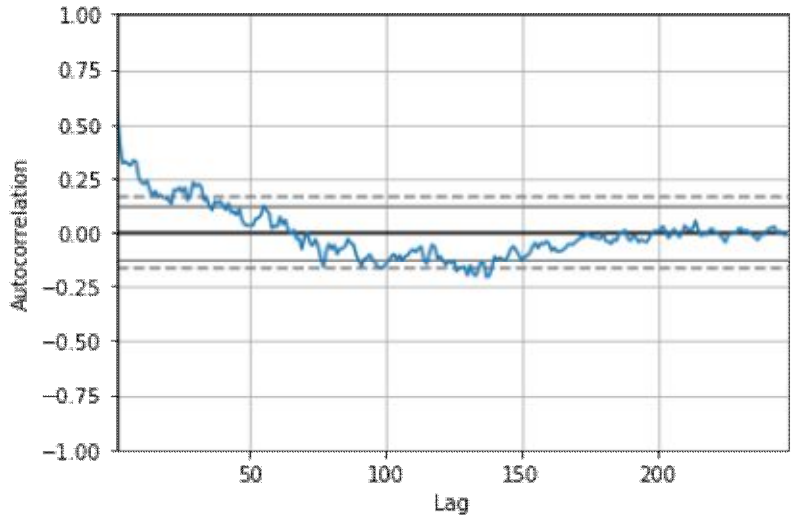


In [59]:

```
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: FutureWarning: 'p
andas.tools.plotting.autocorrelation_plot' is deprecated, import 'pandas.plotting.autocorre
lation_plot' instead.
```

Out[59]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x204f27b6630>
```



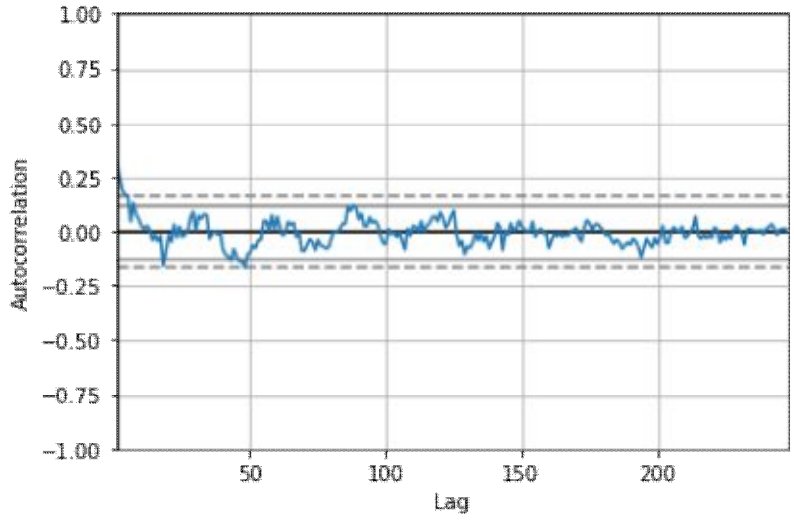
In [60]:

```
autocorrelation_plot(np.log(nse_tcs['Volume']))
```

```
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: FutureWarning: 'p
andas.tools.plotting.autocorrelation_plot' is deprecated, import 'pandas.plotting.autocorre
lation_plot' instead.
  """Entry point for launching an IPython kernel.
```

Out[60]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x204ef8d1c50>
```

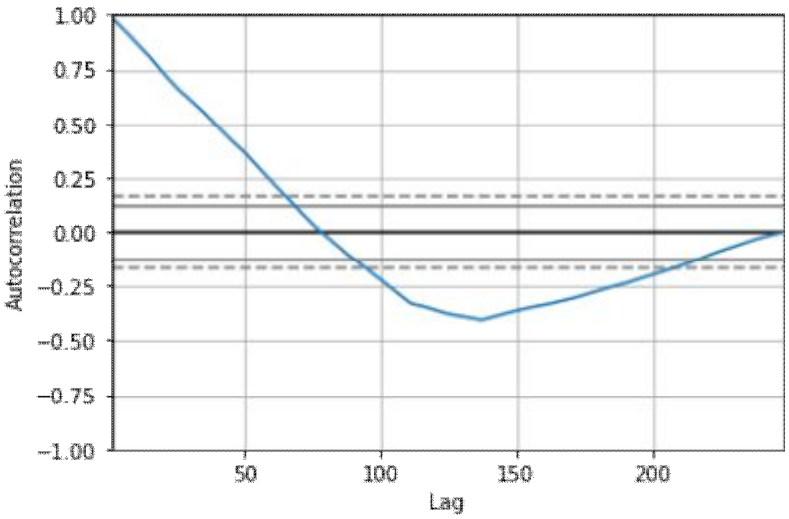


In [61]:

```
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: FutureWarning: 'p
andas.tools.plotting.autocorrelation_plot' is deprecated, import 'pandas.plotting.autocorre
lation_plot' instead.
  """Entry point for launching an IPython kernel.
```

Out[61]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x204f2888d68>
```

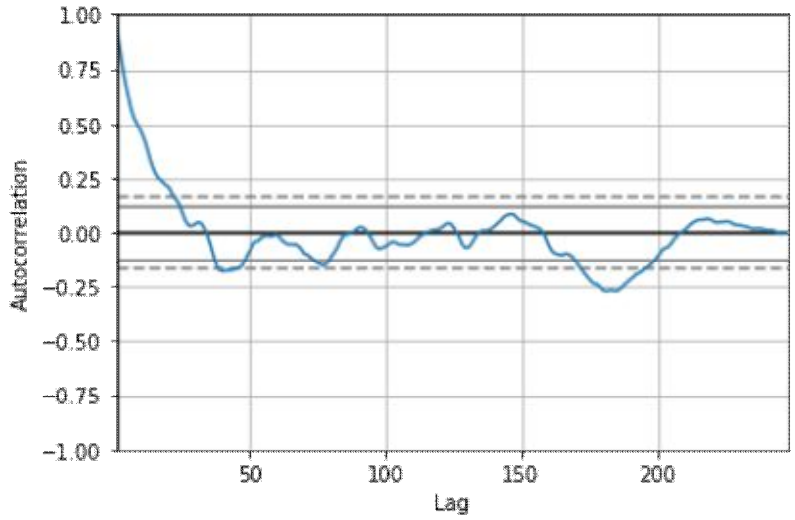


In [62]:

```
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: FutureWarning: 'p
andas.tools.plotting.autocorrelation_plot' is deprecated, import 'pandas.plotting.autocorre
lation_plot' instead.
  """Entry point for launching an IPython kernel.
```

Out[62]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x204f28d5c88>
```



In [63]:

```
nse_infy=nse_infy.drop(['T','notvolshock','PreviousVolume','T1'],axis=1).head()
```

In [64]:

```
nse_tcs=nse_tcs.drop(['T','notvolshock','PreviousVolume','T1'],axis=1).head()
```

In [65]:

```
x=nse_infy.drop(['Close','Symbol','Series'],axis=1)
y=nse_infy['Close']
```

In [66]:

```
import sklearn.model_selection as model_selection
x_train,x_test,y_train,y_test=model_selection.train_test_split(x,y,test_size=0.20,random_st
ate=200)
```

In [67]:

```
import sklearn.linear_model as linear_model
```

In [68]:

```
reg=linear_model.Ridge(normalize=True,fit_intercept=True)
reg=reg.fit(x_train,y_train)
```

In [69]:

Out[69]:

```
{'alpha': 99.1}
```

In [70]:

```
reg=linear_model.Ridge(normalize=True,fit_intercept=True,alpha=99.1)
reg=reg.fit(x_train,y_train)
reg.coef_
```

Out[70]:

```
array([ 3.95348801e-03,  3.44343167e-03,  6.40849456e-03,  1.36441625e-02,
        8.41724573e-03,  8.69453162e-03,  7.38063198e-08,  3.88842417e-16,
        1.72158077e-06,  9.74845159e-08,  6.91166204e-01,  1.65754154e-02,
        1.65754154e-02,  1.65754154e-02,  1.65754154e-02,  1.65754154e-02,
       -3.20000526e-01, -3.20385487e-02, -3.20000526e-01,  9.29126429e-02])
```

In [71]:

```
reg.intercept_
```

Out[71]:

```
1733.3273414111247
```

In [72]:

```
import sklearn.metrics as metrics
import sklearn.preprocessing as preprocessing
```

In [73]:

```
x_test=preprocessing.normalize(x_test)
```

In [76]:

```
print('MAE:',metrics.mean_squared_error(y_test,greg.predict(x_test)))
```

```
MAE: 48784.73131211767
```

In [80]:

```
x1=nse_tcs.drop(['Close','Symbol','Series'],axis=1)
y1=nse_tcs['Close']
x1_train,x1_test,y1_train,y1_test=model_selection.train_test_split(x1,y1,test_size=0.20,ran
dom_state=200)
```

In [81]:

```
reg1=linear_model.Lasso(max_iter=1000,normalize=True)
reg1=reg1.fit(x1_train,y1_train)
```

In [82]:

```
greg1=model_selection.GridSearchCV(reg1,param_grid={'alpha':np.arange(0.1,100,1).tolist()})
greg1=greg1.fit(x1_train,y1_train)
greg1.best_params_
```

```
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the numbe
r of iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the numbe
r of iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the numbe
r of iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the numbe
r of iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the numbe
r of iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the numbe
r of iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the numbe
r of iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the numbe
r of iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
C:\Users\TINU ROHITH\Anaconda3\lib\site-packages\sklearn\linear_model\coordinate_descent.p
y:491: ConvergenceWarning: Objective did not converge. You might want to increase the numbe
r of iterations. Fitting data with very small alpha may cause precision problems.
  ConvergenceWarning)
```

Out[82]:

```
{'alpha': 14.1}
```

In [83]:

```
reg1=linear_model.Lasso(max_iter=1000,normalize=True,alpha=14.1)
reg1=reg1.fit(x1_train,y1_train)
reg1.coef_
```

Out[83]:

```
array([ 0.        ,  0.        ,  0.        ,  0.0610708 ,  0.4857956 ,
        0.00667997, -0.        , -0.        , -0.        , -0.        ,
       -0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        , -0.        , -0.        , -0.        ,  0.        ])
```

In [84]:

```
reg1.intercept_
```

Out[84]:

```
1123.7605381386804
```

In [85]:

```
x1_test=preprocessing.normalize(x1_test)
```

In [86]:

```
print('MAE:',metrics.mean_squared_error(y1_test,greg1.predict(x1_test)))
```

```
MAE: 1749904.2418575333
```

In [87]:

```
import sklearn.tree as tree
from sklearn.ensemble import GradientBoostingClassifier
x2=nse_infy.drop(['VolumeShocks','Symbol','Series'],axis=1)
x2=pd.get_dummies(x2)
y2=nse_infy['VolumeShocks']
x2_train,x2_test,y2_train,y2_test=model_selection.train_test_split(x2,y2,test_size=0.20,ran
dom_state=200)
```

In [88]:

```
clf=GradientBoostingClassifier(n_estimators=80,random_state=400)
clf.fit(x2_train,y2_train)
```

Out[88]:

```
GradientBoostingClassifier(criterion='friedman_mse', init=None,
              learning_rate=0.1, loss='deviance', max_depth=3,
              max_features=None, max_leaf_nodes=None,
              min_impurity_decrease=0.0, min_impurity_split=None,
              min_samples_leaf=1, min_samples_split=2,
              min_weight_fraction_leaf=0.0, n_estimators=80,
              presort='auto', random_state=400, subsample=1.0, verbose=0,
              warm_start=False)
```

In [89]:

```
clf.score(x2_test,y2_test)
```

Out[89]:

```
1.0
```

In [90]:

```
clf.feature_importances_
```

Out[90]:

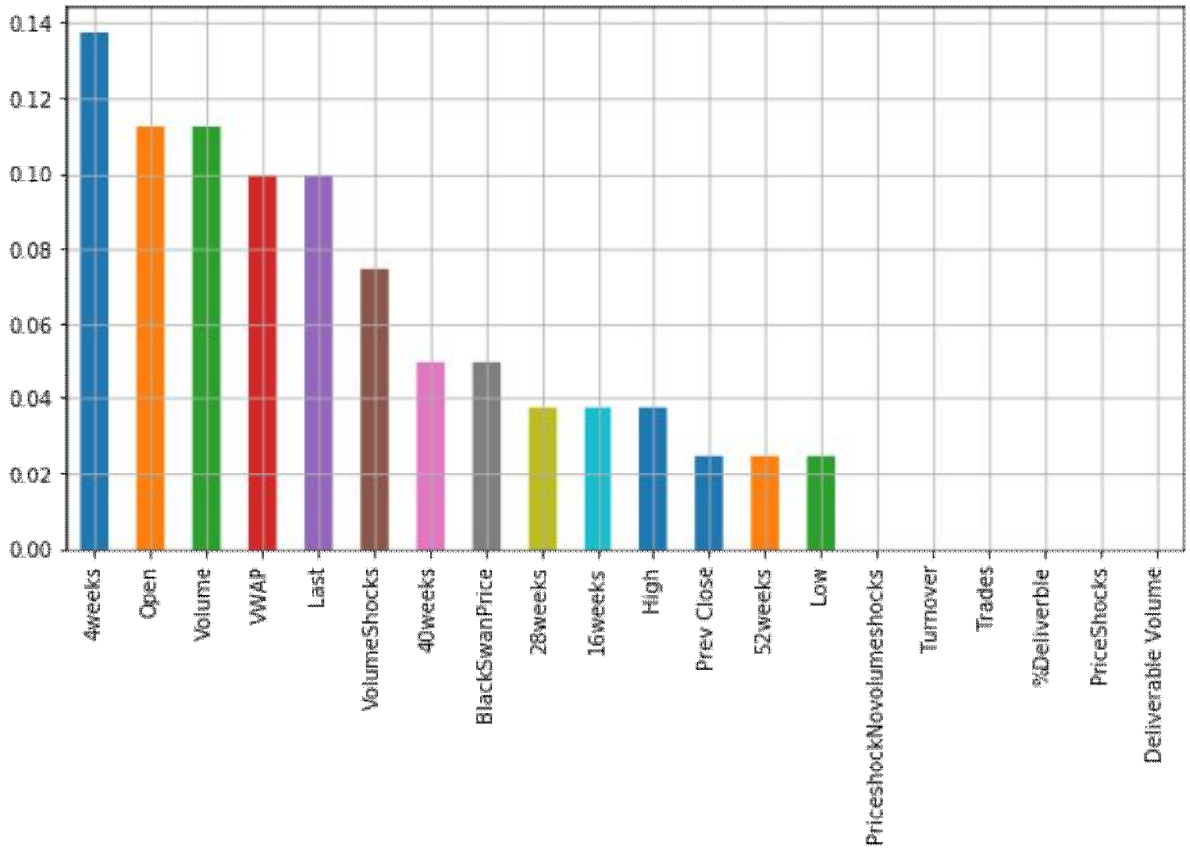
```
array([0.025 , 0.1125, 0.0375, 0.025 , 0.1   , 0.1   , 0.1125, 0.    ,
       0.    , 0.    , 0.    , 0.1375, 0.0375, 0.0375, 0.05  , 0.025 ,
       0.075 , 0.    , 0.05  , 0.    ])
```

In [91]:

```
feature=pd.Series(clf.feature_importances_,index=x.columns)
feature.sort_values(ascending=False)
feature.sort_values(ascending=False).plot(kind='bar',figsize=(10,5),grid=True)
```

Out[91]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x204f5762a20>
```



In [92]:

```
x3=nse_tcs.drop(['VolumeShocks','Symbol','Series'],axis=1)
x3=pd.get_dummies(x3)
y3=nse_tcs['VolumeShocks']
x3_train,x3_test,y3_train,y3_test=model_selection.train_test_split(x3,y3,test_size=0.20,ran
dom_state=200)
```

In [93]:

```
clf1=GradientBoostingClassifier(n_estimators=80,random_state=400)
clf1.fit(x3_train,y3_train)
```

Out[93]:

```
GradientBoostingClassifier(criterion='friedman_mse', init=None,
              learning_rate=0.1, loss='deviance', max_depth=3,
              max_features=None, max_leaf_nodes=None,
              min_impurity_decrease=0.0, min_impurity_split=None,
              min_samples_leaf=1, min_samples_split=2,
              min_weight_fraction_leaf=0.0, n_estimators=80,
              presort='auto', random_state=400, subsample=1.0, verbose=0,
              warm_start=False)
```

In [94]:

```
clf1.score(x3_test,y3_test)
```

Out[94]:

```
0.0
```

In [95]:

```
clf1.feature_importances_
```

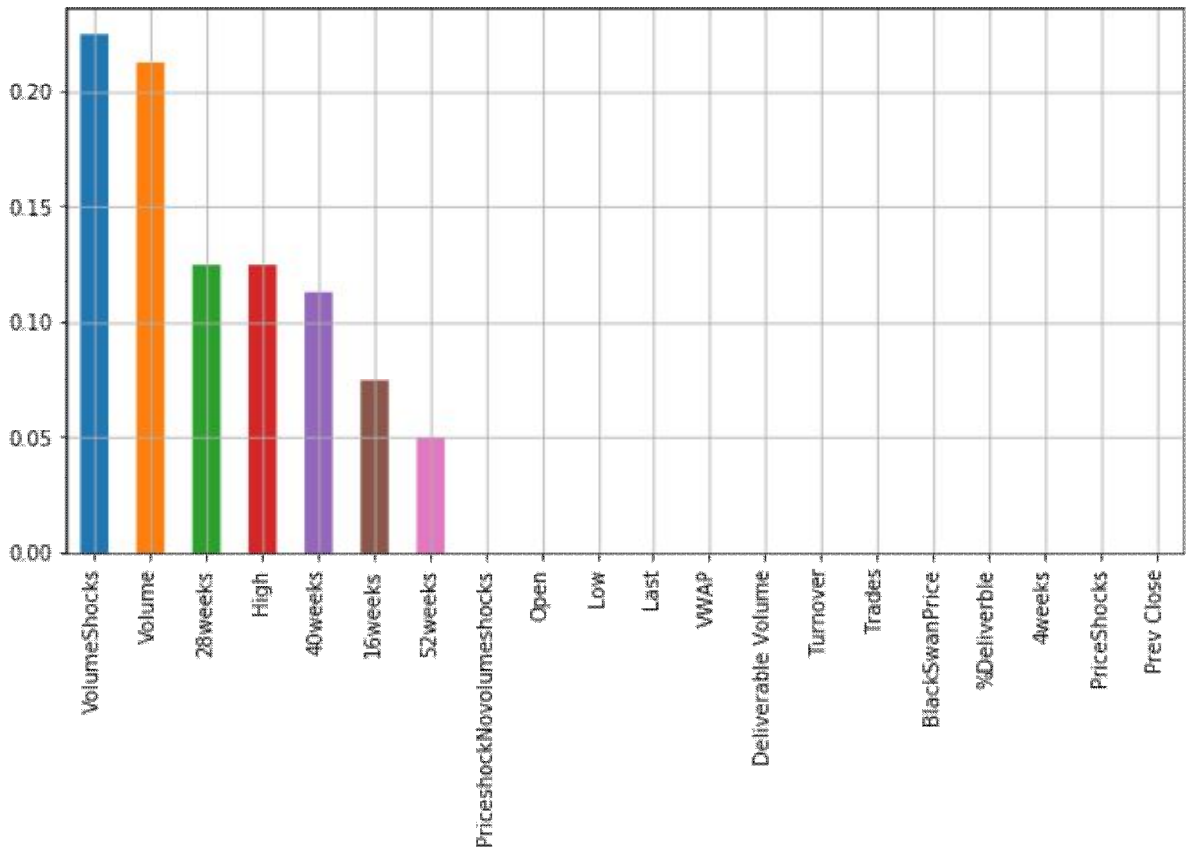
Out[95]:

```
array([0.    , 0.    , 0.125 , 0.    , 0.    , 0.    , 0.2125, 0.    ,
       0.    , 0.    , 0.    , 0.    , 0.075 , 0.125 , 0.1125, 0.05  ,
       0.225 , 0.    , 0.    , 0.    ])
```

In [98]:

Out[98]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x204f59cc7f0>
```


0.20
0.15
0.10
0.05
0.00
VolumeShocks
Volume
28weeks
High
40weeks
16weeks
52weeks
PriceshockNovolumeshocks
Open
Low
Last
VWAP
Deliverable Volume
Turnover
Trades
BlackSwanPrice
%Deliverble
4weeks
PriceShocks
Prev Close