In [33]:
```
import pandas as pd
import numpy as np
import os
```

In [34]:
```
os.chdir("/home/prathikm/Desktop/ds_assignments/in_class_notebook_and_csvs/Normal_Dist
```

In [35]:
```
beml_df = pd.read_csv("BEML.csv")
beml_df.head(5)
```

Out[35]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2010-01-04	1121.0	1151.00	1121.00	1134.0	1135.60	101651.0	1157.18
1	2010-01-05	1146.8	1149.00	1128.75	1135.0	1134.60	59504.0	676.47
2	2010-01-06	1140.0	1164.25	1130.05	1137.0	1139.60	128908.0	1482.84
3	2010-01-07	1142.0	1159.40	1119.20	1141.0	1144.15	117871.0	1352.98
4	2010-01-08	1156.0	1172.00	1140.00	1141.2	1144.05	170063.0	1971.42

In [36]:
```
glaxo_df = pd.read_csv("GLAXO.csv")
glaxo_df.head(5)
```

Out[36]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2010-01-04	1613.00	1629.10	1602.00	1629.0	1625.65	9365.0	151.74
1	2010-01-05	1639.95	1639.95	1611.05	1620.0	1616.80	38148.0	622.58
2	2010-01-06	1618.00	1644.00	1617.00	1639.0	1638.50	36519.0	595.09
3	2010-01-07	1645.00	1654.00	1636.00	1648.0	1648.70	12809.0	211.00
4	2010-01-08	1650.00	1650.00	1626.55	1640.0	1639.80	28035.0	459.11

In [37]:
```
#we need date and close price to calulcate the gain - which is what we are interested
beml_df = beml_df[['Date', 'Close']]
glaxo_df = glaxo_df[['Date', 'Close']]
```

In [38]:
```
beml_df
```

Out[38]:

	Date	Close
0	2010-01-04	1135.60
1	2010-01-05	1134.60
2	2010-01-06	1139.60
3	2010-01-07	1144.15
4	2010-01-08	1144.05
...	...	...
1734	2016-12-26	950.25
1735	2016-12-27	975.70
1736	2016-12-28	974.40
1737	2016-12-29	986.05

	Date	Close
1738	2016-12-30	1000.60

1739 rows × 2 columns

In [39]:
```
#beml_df = pd.DatetimeIndex(beml_df['Date'])
#beml_df
'''The DataFrames have a date column, so we can
create a DatetimeIndex index from this column Date. It will ensure that the rows are s
ascending order.'''
```

Out[39]:
```
'The DataFrames have a date column, so we can\ncreate a DatetimeIndex index from this c
olumn Date. It will ensure that the rows are sorted by time in\nascending order.'
```

In [40]:
```
'''The DataFrames have a date column, so we can
create a DatetimeIndex index from this column Date. It will ensure that the rows are s
ascending order.'''
glaxo_df = glaxo_df.set_index(pd.DatetimeIndex(glaxo_df['Date']))
beml_df = beml_df.set_index(pd.DatetimeIndex(beml_df['Date']))
#make date column as
```

https://stackoverflow.com/questions/27032052/how-do-i-properly-set-the-datetimeindex-for-a-pandas-datetime-object-in-a-datafr

In [41]:
```
glaxo_df #quick check at the modified index
```

Out[41]:

	Date	Close
Date		
2010-01-04	2010-01-04	1625.65
2010-01-05	2010-01-05	1616.80
2010-01-06	2010-01-06	1638.50
2010-01-07	2010-01-07	1648.70
2010-01-08	2010-01-08	1639.80
...	...	...
2016-12-26	2016-12-26	2723.50
2016-12-27	2016-12-27	2701.75
2016-12-28	2016-12-28	2702.15
2016-12-29	2016-12-29	2727.90
2016-12-30	2016-12-30	2729.80

1739 rows × 2 columns

In [11]:
```
beml_df
```

Out[11]:

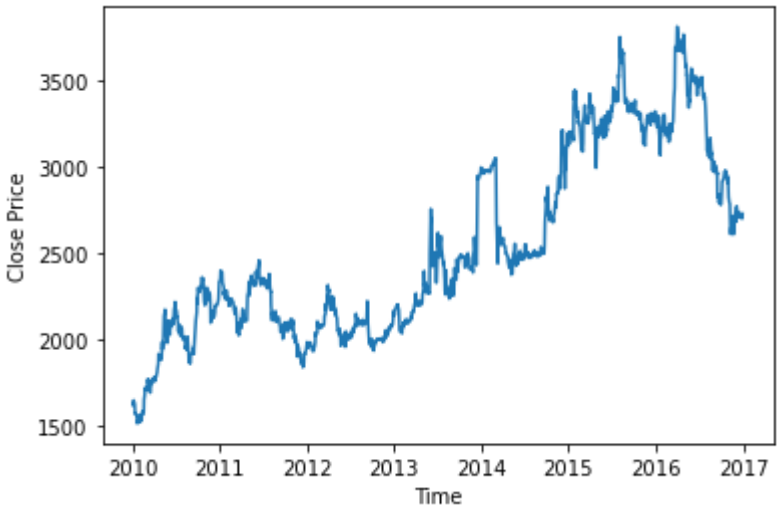
	Date	Close
Date		
2010-01-04	2010-01-04	1135.60
2010-01-05	2010-01-05	1134.60
2010-01-06	2010-01-06	1139.60

	Date	Close
Date		
2010-01-07	2010-01-07	1144.15
2010-01-08	2010-01-08	1144.05
...	...	...
2016-12-26	2016-12-26	950.25
2016-12-27	2016-12-27	975.70
2016-12-28	2016-12-28	974.40
2016-12-29	2016-12-29	986.05
2016-12-30	2016-12-30	1000.60

1739 rows × 2 columns

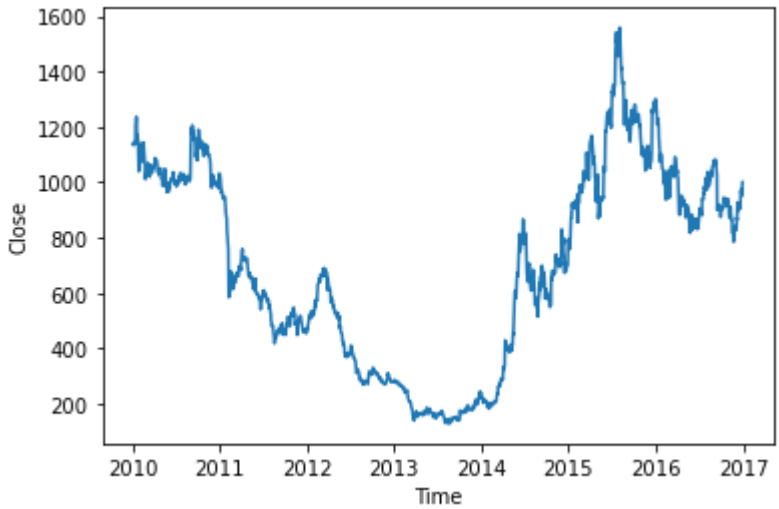
In [12]:

```
import matplotlib.pyplot as plt
import seaborn as sn
%matplotlib inline
plt.plot(glaxo_df.Close);
plt.xlabel('Time');
plt.ylabel('Close Price'); #plot the close price with respect to index i.e. date
```



In [13]:

```
plt.plot(beml_df.Close);
plt.xlabel('Time');
plt.ylabel('Close');
```



In [14]:
```
#compute the gain, with pct() percentage change and period as 1 for one previous day t
glaxo_df['gain'] = glaxo_df.Close.pct_change(periods = 1)
beml_df['gain'] = beml_df.Close.pct_change(periods = 1)
```

In [15]:
```
glaxo_df
```

Out[15]:

Date	Date	Close	gain
2010-01-04	2010-01-04	1625.65	NaN
2010-01-05	2010-01-05	1616.80	-0.005444
2010-01-06	2010-01-06	1638.50	0.013422
2010-01-07	2010-01-07	1648.70	0.006225
2010-01-08	2010-01-08	1639.80	-0.005398
...	...	...	...
2016-12-26	2016-12-26	2723.50	-0.001283
2016-12-27	2016-12-27	2701.75	-0.007986
2016-12-28	2016-12-28	2702.15	0.000148
2016-12-29	2016-12-29	2727.90	0.009529
2016-12-30	2016-12-30	2729.80	0.000697

1739 rows × 3 columns

In [16]:
```
beml_df
```

Out[16]:

Date	Date	Close	gain
2010-01-04	2010-01-04	1135.60	NaN
2010-01-05	2010-01-05	1134.60	-0.000881
2010-01-06	2010-01-06	1139.60	0.004407
2010-01-07	2010-01-07	1144.15	0.003993
2010-01-08	2010-01-08	1144.05	-0.000087
...	...	...	...
2016-12-26	2016-12-26	950.25	-0.021924
2016-12-27	2016-12-27	975.70	0.026782
2016-12-28	2016-12-28	974.40	-0.001332
2016-12-29	2016-12-29	986.05	0.011956
2016-12-30	2016-12-30	1000.60	0.014756

1739 rows × 3 columns

In [17]:
```
#drop first row since it is NaN
glaxo_df = glaxo_df.dropna()
beml_df = beml_df.dropna()
```

In [18]:
```
glaxo_df
```

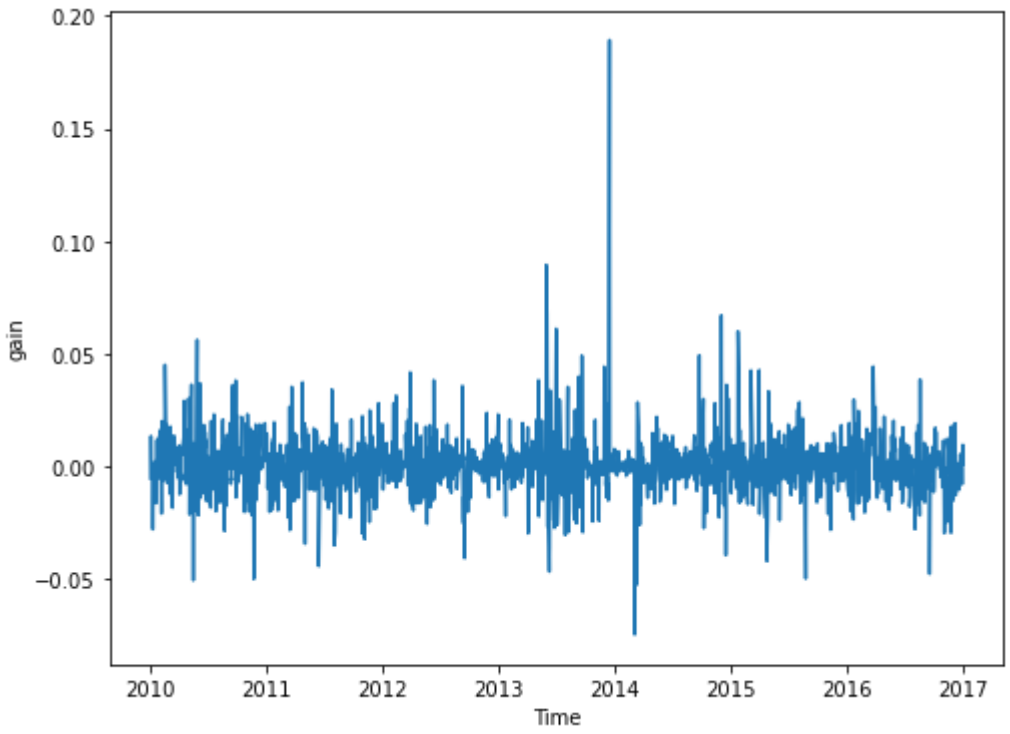
Out[18]:

Date	Date	Close	gain
2010-01-05	2010-01-05	1616.80	-0.005444
2010-01-06	2010-01-06	1638.50	0.013422
2010-01-07	2010-01-07	1648.70	0.006225
2010-01-08	2010-01-08	1639.80	-0.005398
2010-01-11	2010-01-11	1629.45	-0.006312
...	...	...	...
2016-12-26	2016-12-26	2723.50	-0.001283
2016-12-27	2016-12-27	2701.75	-0.007986
2016-12-28	2016-12-28	2702.15	0.000148
2016-12-29	2016-12-29	2727.90	0.009529
2016-12-30	2016-12-30	2729.80	0.000697

1738 rows × 3 columns

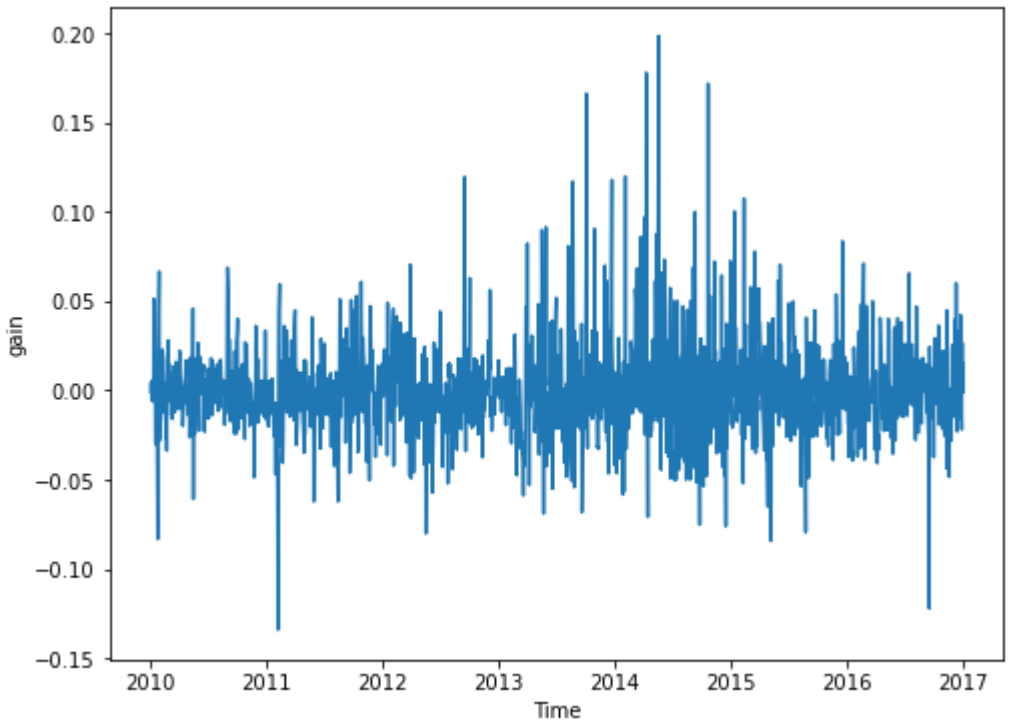
In [19]:

```
#Plot the gains
plt.figure(figsize = (8, 6));
plt.plot(glaxo_df.index, glaxo_df.gain);
plt.xlabel('Time');
plt.ylabel('gain');
```



In [20]:

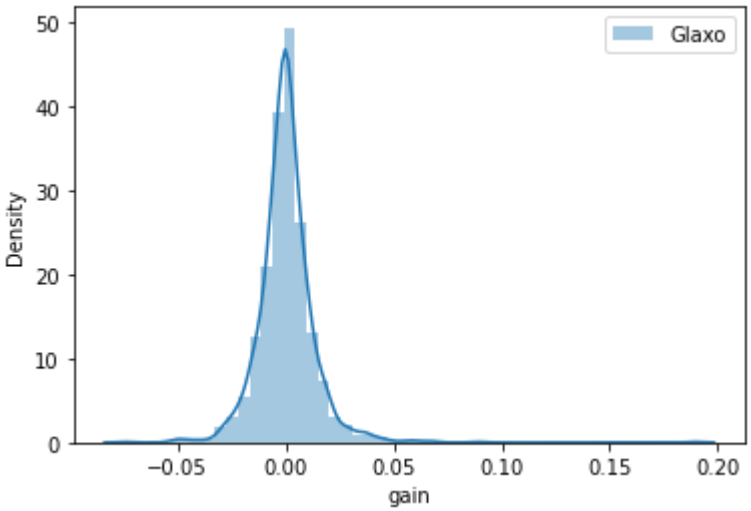
```
#Plot the gains beml
plt.figure(figsize = (8, 6));
plt.plot(beml_df.index, beml_df.gain);
plt.xlabel('Time');
plt.ylabel('gain');
```



In [21]:

```
#density plots
sn.distplot(glaxo_df.gain, label = 'Glaxo');
plt.xlabel('gain');
plt.ylabel('Density');
plt.legend();
```

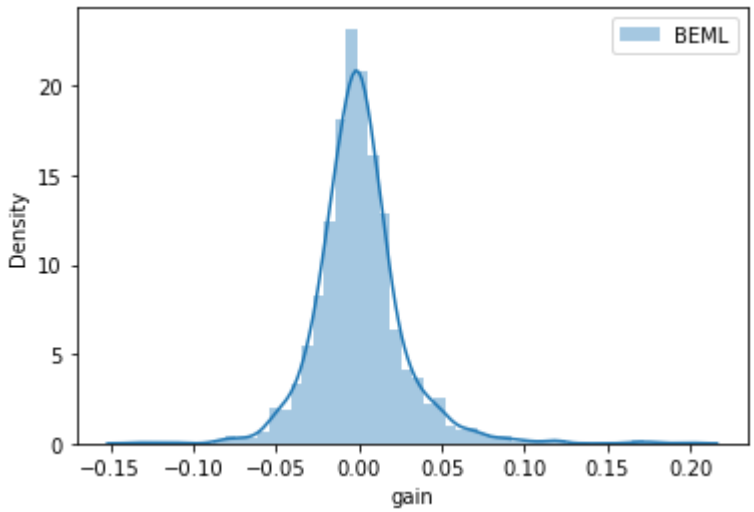
```
/home/prathikm/miniconda3/envs/ds/lib/python3.8/site-packages/seaborn/distributions.py:
2551: FutureWarning: `distplot` is a deprecated function and will be removed in a futur
e version. Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [22]:

```
sn.distplot(beml_df.gain, label = 'BEML');
plt.xlabel('gain');
plt.ylabel('Density');
plt.legend();
```

```
/home/prathikm/miniconda3/envs/ds/lib/python3.8/site-packages/seaborn/distributions.py:
2551: FutureWarning: `distplot` is a deprecated function and will be removed in a futur
e version. Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [23]:
```
print('Mean:', round(glaxo_df.gain.mean(), 4))
print('Standard Deviation: ', round(glaxo_df.gain.std(), 4))
```

```
Mean: 0.0004
Standard Deviation:  0.0134
```

In [24]:
```
print('Mean: ', round(beml_df.gain.mean(), 4))
print('Standard Deviation: ', round(beml_df.gain.std(), 4))
```

```
Mean:  0.0003
Standard Deviation:  0.0264
```

Compute 2% loss or gain for Glaxo

In [25]:
```
from scipy import stats
#Probability of making 2% loss or higher in Glaxo
stats.norm.cdf( -0.02,
loc=glaxo_df.gain.mean(),
scale=glaxo_df.gain.std())
```

Out[25]: 0.06352488667177401

In [26]:
```
#Probability of making 2% gain or higher in Glaxo
1 - stats.norm.cdf(0.02,
loc=glaxo_df.gain.mean(),
scale=glaxo_df.gain.std())
```

Out[26]: 0.07104511457618568

Compute 2% loss or gain for BEML

In [27]:
```
#Probability of making 2% loss or higher in BEML
stats.norm.cdf( -0.02,
loc=beml_df.gain.mean(),
scale=beml_df.gain.std())
```

Out[27]: 0.22155987503755292

In [63]:
```
#Probability of making 2% gain or higher in BEML
1 - stats.norm.cdf(0.02,
loc=beml_df.gain.mean(),
scale=beml_df.gain.std())
```

Out[63]: 0.22769829484075355