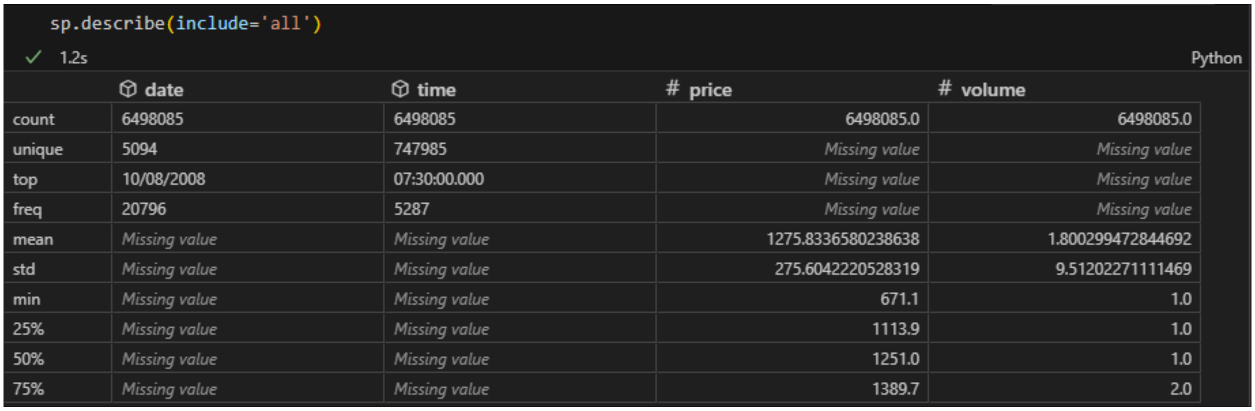
1. Use S&P 500 futures tick data taken from

https://www.kaggle.com/datasets/finnhub/sp-500-futures-tick-data-sp. Save it as SP.csv. Remove the rows with 0 volume and then proceed.



```
sp.describe(include='all')
```

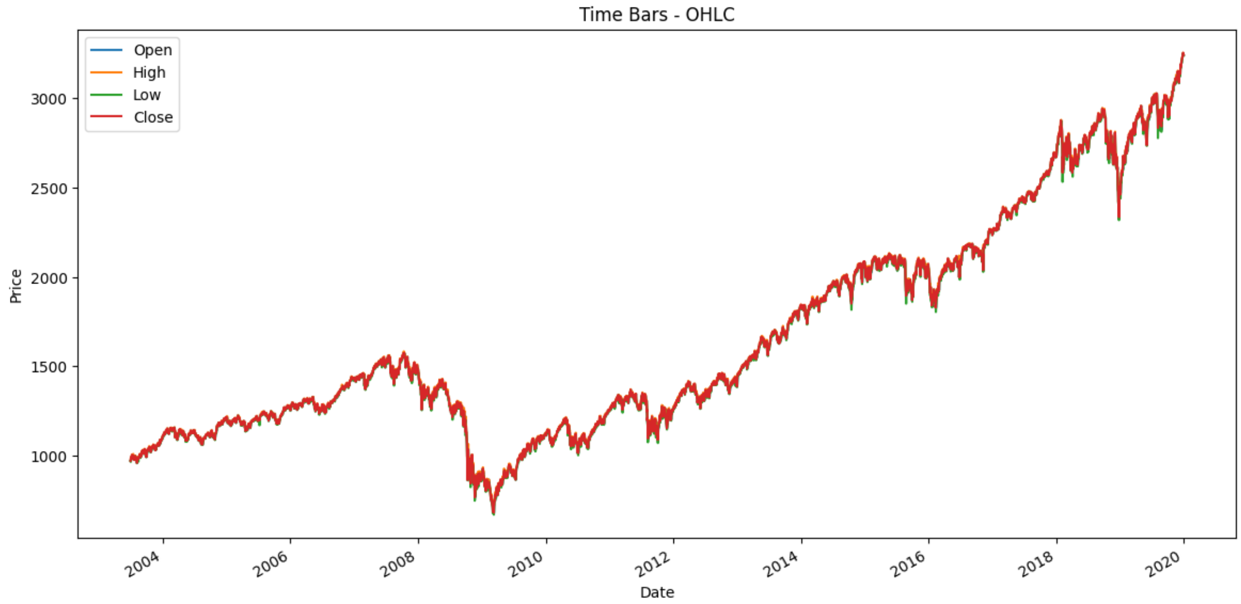
✓ 1.2s Python

	date	time	price	volume
count	6498085	6498085	6498085.0	6498085.0
unique	5094	747985	Missing value	Missing value
top	10/08/2008	07:30:00.000	Missing value	Missing value
freq	20796	5287	Missing value	Missing value
mean	Missing value	Missing value	1275.8336580238638	1.800299472844692
std	Missing value	Missing value	275.6042220528319	9.51202271111469
min	Missing value	Missing value	671.1	1.0
25%	Missing value	Missing value	1113.9	1.0
50%	Missing value	Missing value	1251.0	1.0
75%	Missing value	Missing value	1389.7	2.0

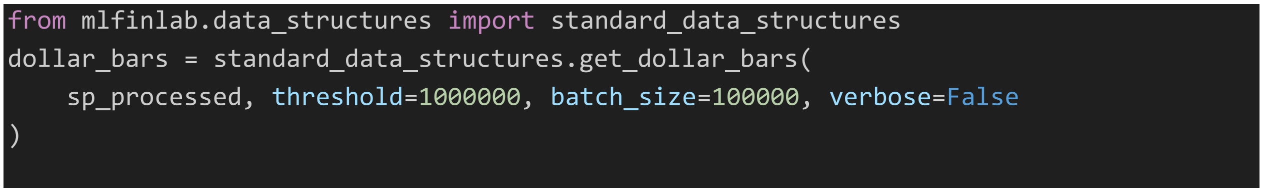
```
# Convert time column to string format first, then combine
sp['date_time'] = pd.to_datetime(sp['date'].astype(str) + ' ' + sp['time'].astype(str))
```

```
sp_processed = sp[['date_time', 'price', 'volume']].copy()
sp_processed.columns = ['date_time', 'price', 'volume']
```

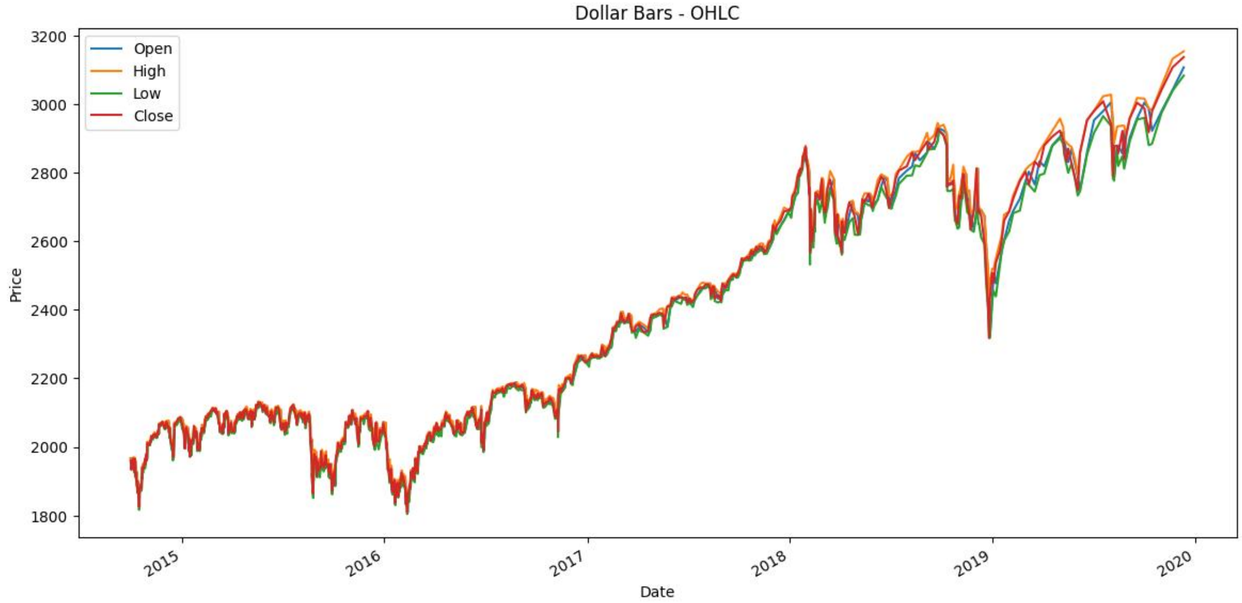
```
from mlfinlab.data_structures import time_data_structures

time_bars = time_data_structures.get_time_bars(
    sp_processed, resolution="D", verbose=False
)
```



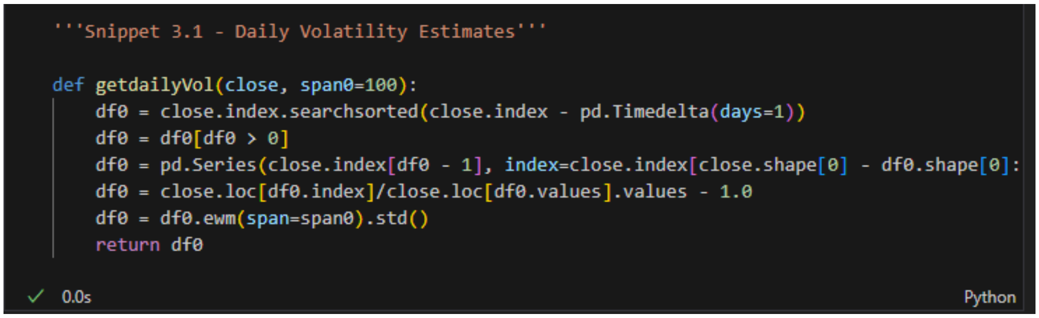
2. Form dollar bars for the data from Exercise 1 above.



```
from mlfinlab.data_structures import standard_data_structures
dollar_bars = standard_data_structures.get_dollar_bars(
    sp_processed, threshold=1000000, batch_size=100000, verbose=False
)
```

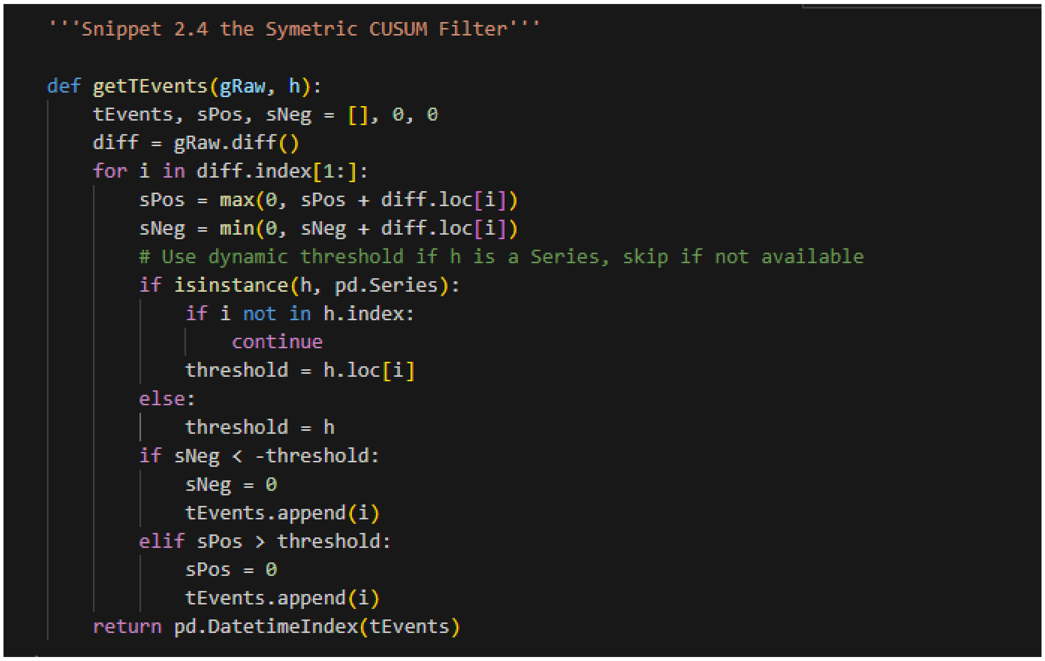


(a) Apply a symmetric CUSUM filter (Chapter 2, Section 2.5.2.1) where the threshold is the standard deviation of daily returns (Snippet 3.1).

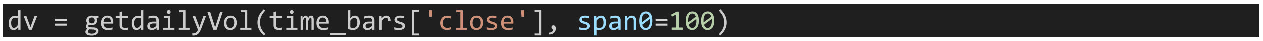


```python
'''Snippet 3.1 - Daily Volatility Estimates'''

def getdailyVol(close, span0=100):
    df0 = close.index.searchsorted(close.index - pd.Timedelta(days=1))
    df0 = df0[df0 > 0]
    df0 = pd.Series(close.index[df0 - 1], index=close.index[close.shape[0] - df0.shape[0]:
    df0 = close.loc[df0.index]/close.loc[df0.values].values - 1.0
    df0 = df0.ewm(span=span0).std()
    return df0
```

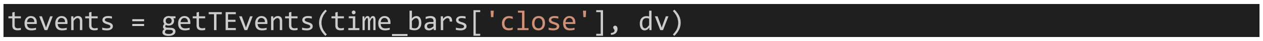
Snippet 2.4 modified to take the series of emwa vols



```python
'''Snippet 2.4 the Symetric CUSUM Filter'''

def getTEvents(gRaw, h):
    tEvents, sPos, sNeg = [], 0, 0
    diff = gRaw.diff()
    for i in diff.index[1:]:
        sPos = max(0, sPos + diff.loc[i])
        sNeg = min(0, sNeg + diff.loc[i])
        # Use dynamic threshold if h is a Series, skip if not available
        if isinstance(h, pd.Series):
            if i not in h.index:
                continue
            threshold = h.loc[i]
        else:
            threshold = h
        if sNeg < -threshold:
            sNeg = 0
            tEvents.append(i)
        elif sPos > threshold:
            sPos = 0
            tEvents.append(i)
    return pd.DatetimeIndex(tEvents)
```



```python
dv = getdailyVol(time_bars['close'], span0=100)
```

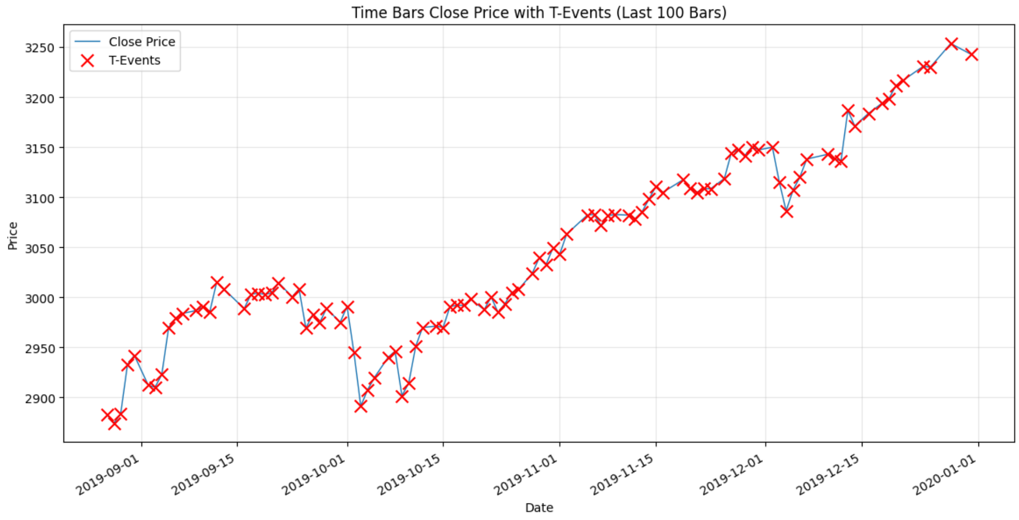


```python
tevents = getTEvents(time_bars['close'], dv)
```

I don’t have a good idea how to align the index of daily emwa vols with Dollar Bars.

I did t events for time bars instead.

Charting time bars vs the T-Events series for the last 100 Time Bars



I have snippets 3.2 Triple Barrier, 3.5 labeling and Size and 3.8 Dropping Under-populated Labels in GitHub

https://github.com/McSavage/MLFinLab/blob/main/notebooks/FIN221_HW3_SP.ipynb

========LOOKS LIKE I'LL NEED A BIT LONGER TO UNWRAP THIS===========

(b) Use Snippet 3.4 on a pandas series t1, where numDays=1.

(c) On those sampled features, apply the triple-barrier method, where ptSl=[1,1]

and t1 is the series you created in part b.

(d) Apply getBins to generate the labels.

3. On data from Exercise 1 above, use Snippet 3.8 to drop rare labels