In [220…
```
import pandas as pd
import numpy as np

from pylab import plt, mpl
plt.style.use('seaborn')
mpl.rcParams['font.family'] = 'serif'
%matplotlib inline
```

In [221…
```
asset_1 = 'bitcoin'
asset_2 = 'tezos'
```

Selecting the dates

In [222…
```
import datetime as dt
'''création des datetimes'''
dt_1 = dt.datetime(2018, 7, 3)
dt_2 = dt.datetime(2021, 8, 7)
dt_1, dt_2
```

Out[222… (datetime.datetime(2018, 7, 3, 0, 0), datetime.datetime(2021, 8, 7, 0, 0))

In [223…
```
'''création des timestamps'''
ts_1 = dt_1.timestamp()
ts_2 = dt_2.timestamp()
ts_1, ts_2
```

Out[223… (1530568800.0, 1628287200.0)

Building an asset class

In [224…
```
import requests

class InfoAsset(object):

    __API_URL_BASE = 'https://api.coingecko.com/api/v3/'

    def __init__(self, id, vs_currency, api_base_url=__API_URL_BASE):
        self.id = id
        self.vs_currency = vs_currency
        self.api_base_url = api_base_url

    def get_coin_market_chart_range_by_id(self, from_timestamp, to_timestamp):
        """Get historical market data include price, market cap, and 24h volume with
        api_url = '{0}coins/{1}/market_chart/range?vs_currency={2}&from={3}&to={4}'.
                                                self.vs_currency, from_timestamp, to_tim
        return requests.get(api_url).json()
```

Getting Asset 1 daily prices

In [225…
```
o = InfoAsset(asset_1, 'usd')
d = o.get_coin_market_chart_range_by_id(ts_1, ts_2)
```

In [226…
```
ts = [elt[0] for elt in d['prices']]
```

```
dt_list = [dt.datetime.fromtimestamp(elt/1000) for elt in ts]
prices = [elt[1] for elt in d['prices']]
data = {asset_1: prices}
df_1 = pd.DataFrame(data = data)
df_1.index = dt_list
df_1.head()
```

Out[226...

	bitcoin
2018-07-03 02:00:00	6595.547313
2018-07-04 02:00:00	6497.607355
2018-07-05 02:00:00	6562.721737
2018-07-06 02:00:00	6537.740872
2018-07-07 02:00:00	6603.294704

In [227...

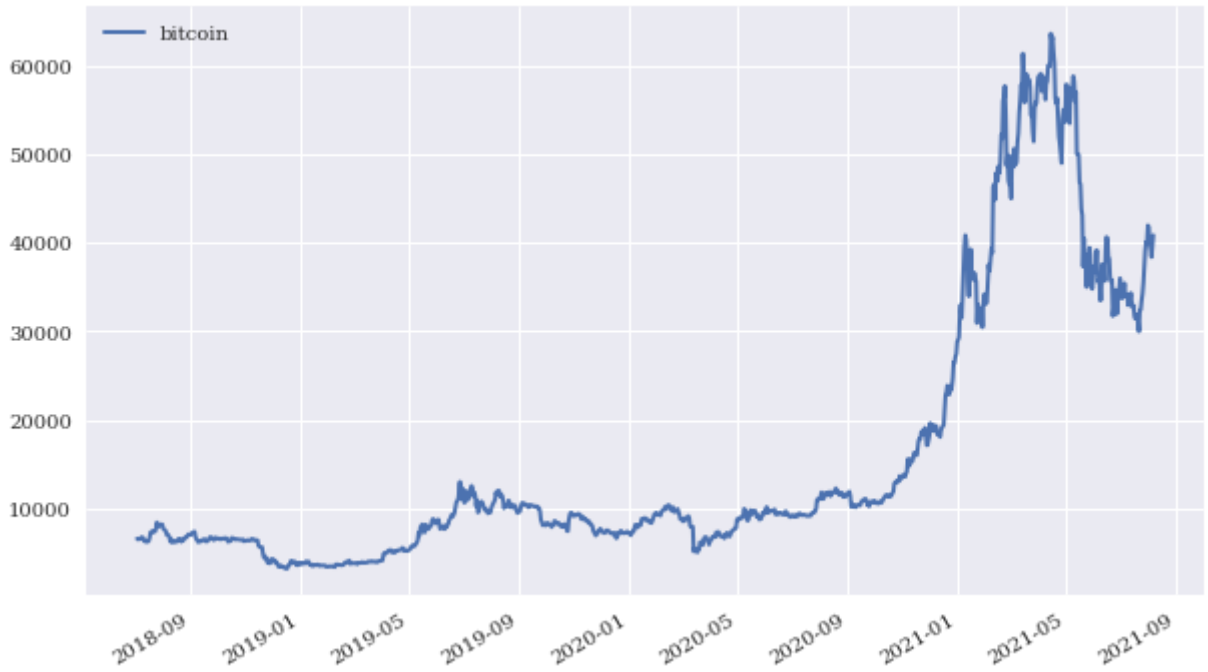
```
df_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1131 entries, 2018-07-03 02:00:00 to 2021-08-06 02:00:00
Data columns (total 1 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   bitcoin  1131 non-null   float64
dtypes: float64(1)
memory usage: 17.7 KB
```

In [228...

```
df_1.plot(lw=2.0, figsize=(10, 6))
```

Out[228... <AxesSubplot:>



Getting Asset 2 daily prices

In [229...

```
o = InfoAsset(asset_2, 'usd')
d = o.get_coin_market_chart_range_by_id(ts_1, ts_2)
```

In [230…
```
ts = [elt[0] for elt in d['prices']]
dt_list = [dt.datetime.fromtimestamp(elt/1000) for elt in ts]
prices = [elt[1] for elt in d['prices']]
data = {asset_2: prices}
df_2 = pd.DataFrame(data = data)
df_2.index = dt_list
df_2.head()
```

Out[230…

	tezos
2018-07-03 02:00:00	2.937866
2018-07-04 02:00:00	2.047985
2018-07-05 02:00:00	1.969392
2018-07-06 02:00:00	1.528033
2018-07-07 02:00:00	1.796304

In [231…
```
df_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1131 entries, 2018-07-03 02:00:00 to 2021-08-06 02:00:00
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   tezos   1131 non-null   float64
dtypes: float64(1)
memory usage: 17.7 KB
```

In [232…
```
df_2.plot(lw=2.0, figsize=(10, 6))
```

Out[232… <AxesSubplot:>

Correlation analysis

The data

In [233…
```
raw = pd.concat([df_1, df_2], axis = 1)
```

```
raw.head()
```

Out[233...

	bitcoin	tezos
2018-07-03 02:00:00	6595.547313	2.937866
2018-07-04 02:00:00	6497.607355	2.047985
2018-07-05 02:00:00	6562.721737	1.969392
2018-07-06 02:00:00	6537.740872	1.528033
2018-07-07 02:00:00	6603.294704	1.796304

In [234...

```
'''We store the data in an .csv file'''
path = r'C:\Users\xxxxx\Documents\Trading\Correlation\data.csv'
raw.to_csv(path, index = True, header = True)
```

In [235...

```
'''We read the data from a .csv file'''
filename = path
raw = pd.read_csv(filename, index_col=0, parse_dates=True)
raw.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1131 entries, 2018-07-03 02:00:00 to 2021-08-06 02:00:00
Data columns (total 2 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   bitcoin  1131 non-null   float64
 1   tezos    1131 non-null   float64
dtypes: float64(2)
memory usage: 26.5 KB
```

In [236...

```
data = raw.dropna()
data.tail()
```

Out[236...

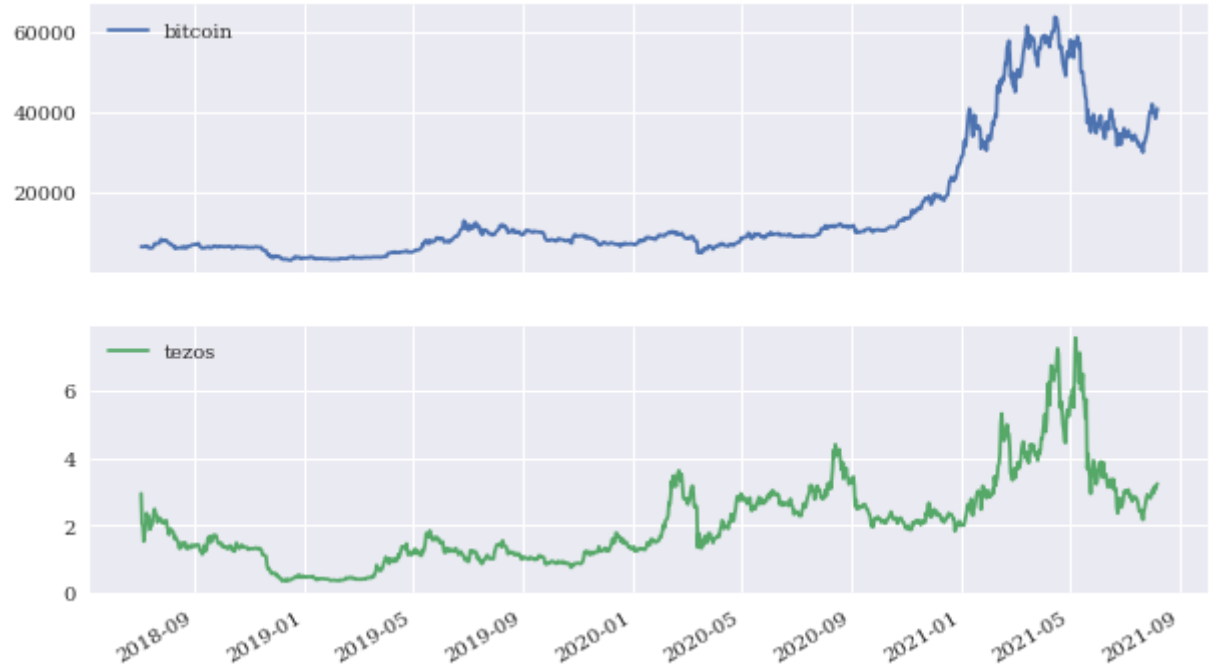
	bitcoin	tezos
2021-08-02 02:00:00	39914.829758	2.949941
2021-08-03 02:00:00	39278.766508	3.152587
2021-08-04 02:00:00	38368.354012	3.067710
2021-08-05 02:00:00	39751.584575	3.181779
2021-08-06 02:00:00	40825.381940	3.230103

In [237...

```
data.plot(subplots=True, figsize=(10, 6))
```

Out[237...

```
array([<AxesSubplot:>, <AxesSubplot:>], dtype=object)
```



In [238...
```
data.plot(secondary_y=asset_2, figsize=(10, 6))
```

Out[238... <AxesSubplot:>

Logarithmic Returns

In [239...
```
rets = np.log(data / data.shift(1))
rets.head()
```

Out[239...

	bitcoin	tezos
2018-07-03 02:00:00	NaN	NaN
2018-07-04 02:00:00	-0.014961	-0.360827
2018-07-05 02:00:00	0.009971	-0.039132
2018-07-06 02:00:00	-0.003814	-0.253744

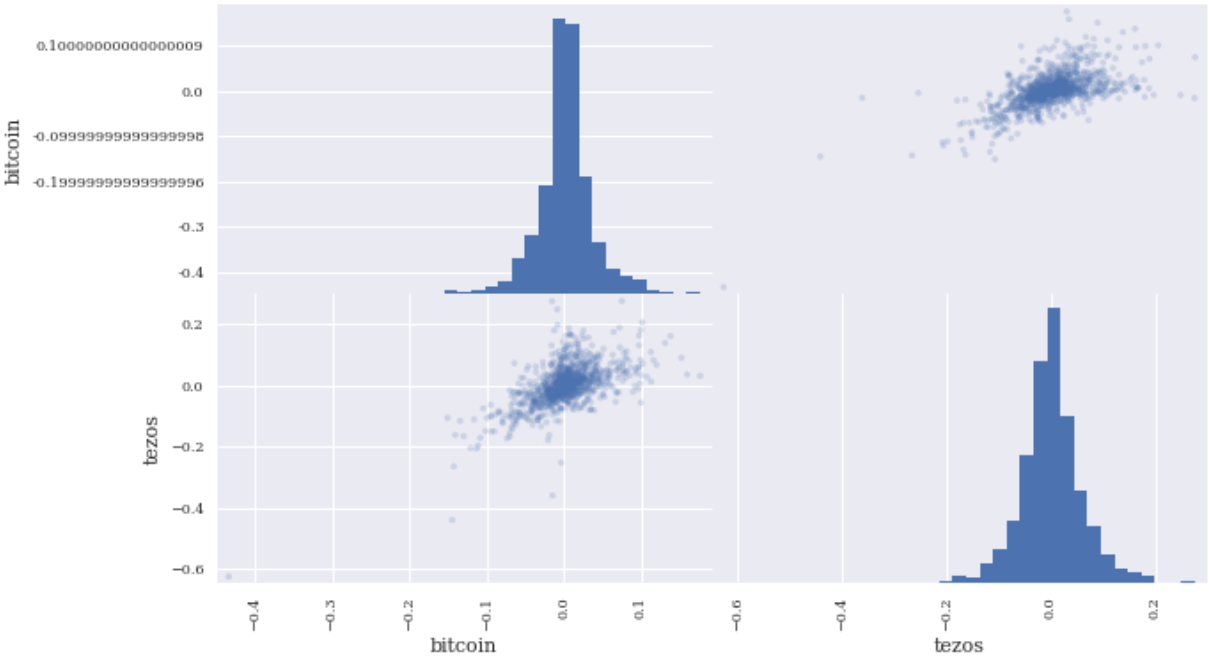
	bitcoin	tezos
2018-07-07 02:00:00	0.009977	0.161750

In [240…
```
rets.dropna(inplace=True)
rets.plot(subplots=True, figsize=(10, 6))
```

Out[240…
```
array([<AxesSubplot:>, <AxesSubplot:>], dtype=object)
```

In [241…
```
pd.plotting.scatter_matrix(rets, alpha=0.2, diagonal='hist',\
                           hist_kwds={'bins': 35}, figsize=(10, 6))
```

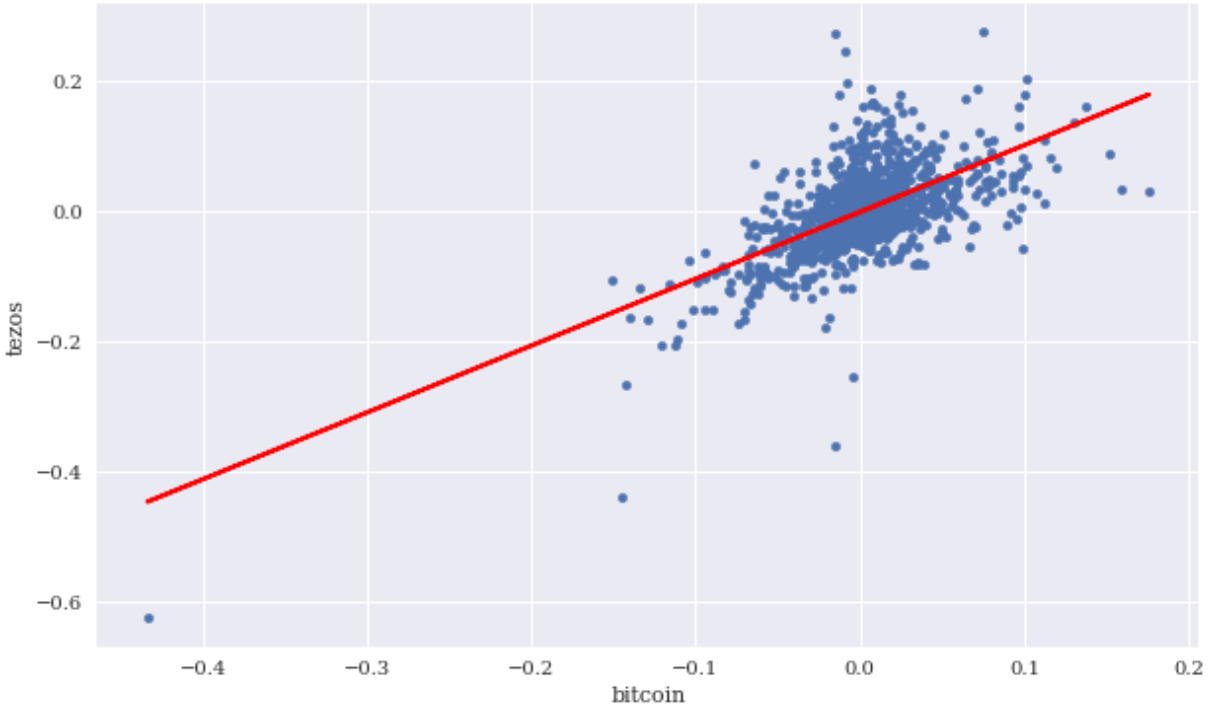
Out[241…
```
array([[<AxesSubplot:xlabel='bitcoin', ylabel='bitcoin'>,
        <AxesSubplot:xlabel='tezos', ylabel='bitcoin'>],
       [<AxesSubplot:xlabel='bitcoin', ylabel='tezos'>,
        <AxesSubplot:xlabel='tezos', ylabel='tezos'>]], dtype=object)
```



OLS Regression

In [242...
```
reg = np.polyfit(rets[asset_1], rets[asset_2], deg=1)
ax = rets.plot(kind='scatter', x=asset_1, y=asset_2, figsize=(10, 6))
ax.plot(rets[asset_1], np.polyval(reg, rets[asset_1]), 'r', lw=2)
```

Out[242...
```
[<matplotlib.lines.Line2D at 0x16db22488d0>]
```



Correlation

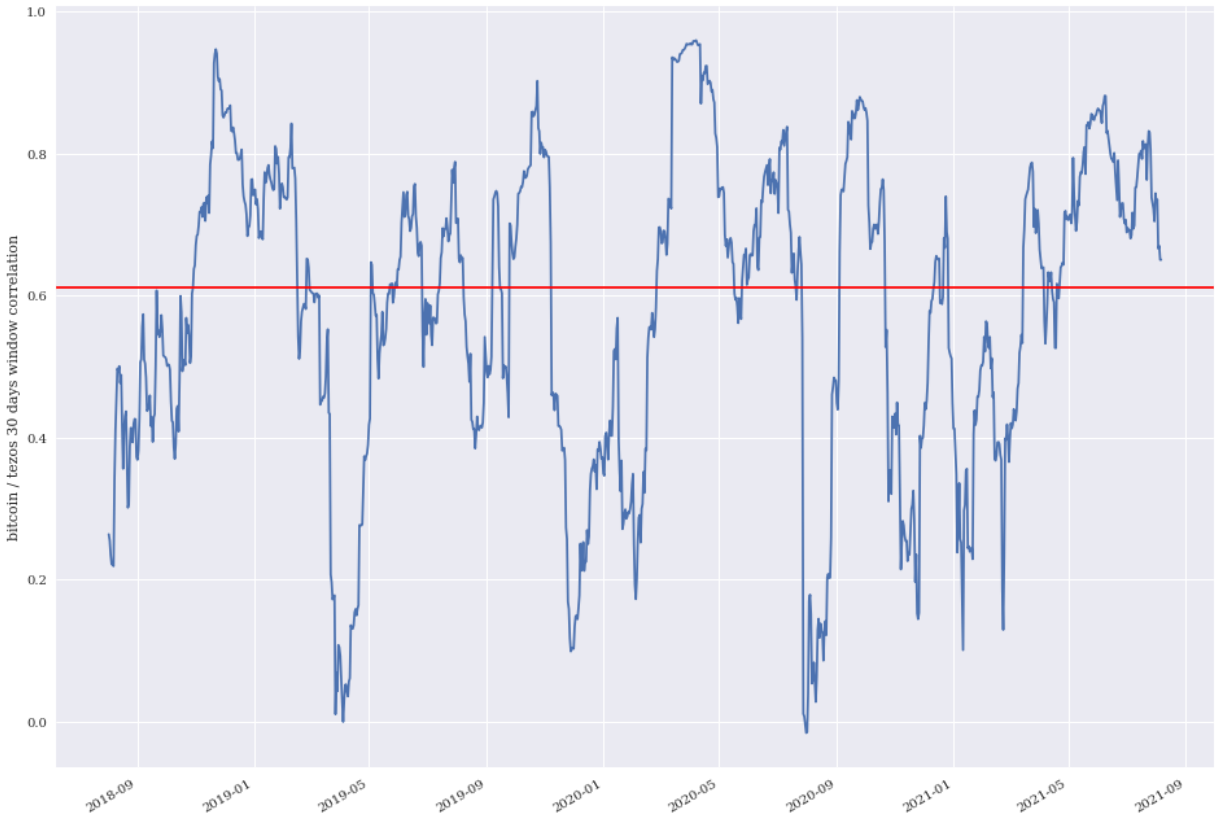
In [243...
```
rets.corr()
```

Out[243...

	bitcoin	tezos
bitcoin	1.000000	0.611075
tezos	0.611075	1.000000

In [244...
```
window = 30
ax = rets[asset_1].rolling(window=window).corr(rets[asset_2]).plot(figsize=(16, 12))
ax.axhline(rets.corr().iloc[0, 1], c='r')
plt.ylabel('{} / {} '.format(asset_1, asset_2)+ str(window) +' days window correlati
```

Out[244...
```
Text(0, 0.5, 'bitcoin / tezos 30 days window correlation')
```


bitcoin / tezos 30 days window correlation
1.0
0.8
0.6
0.4
0.2
0.0
2018-09
2019-01
2019-05
2019-09
2020-01
2020-05
2020-09
2021-01
2021-05
2021-09

In [ ]: