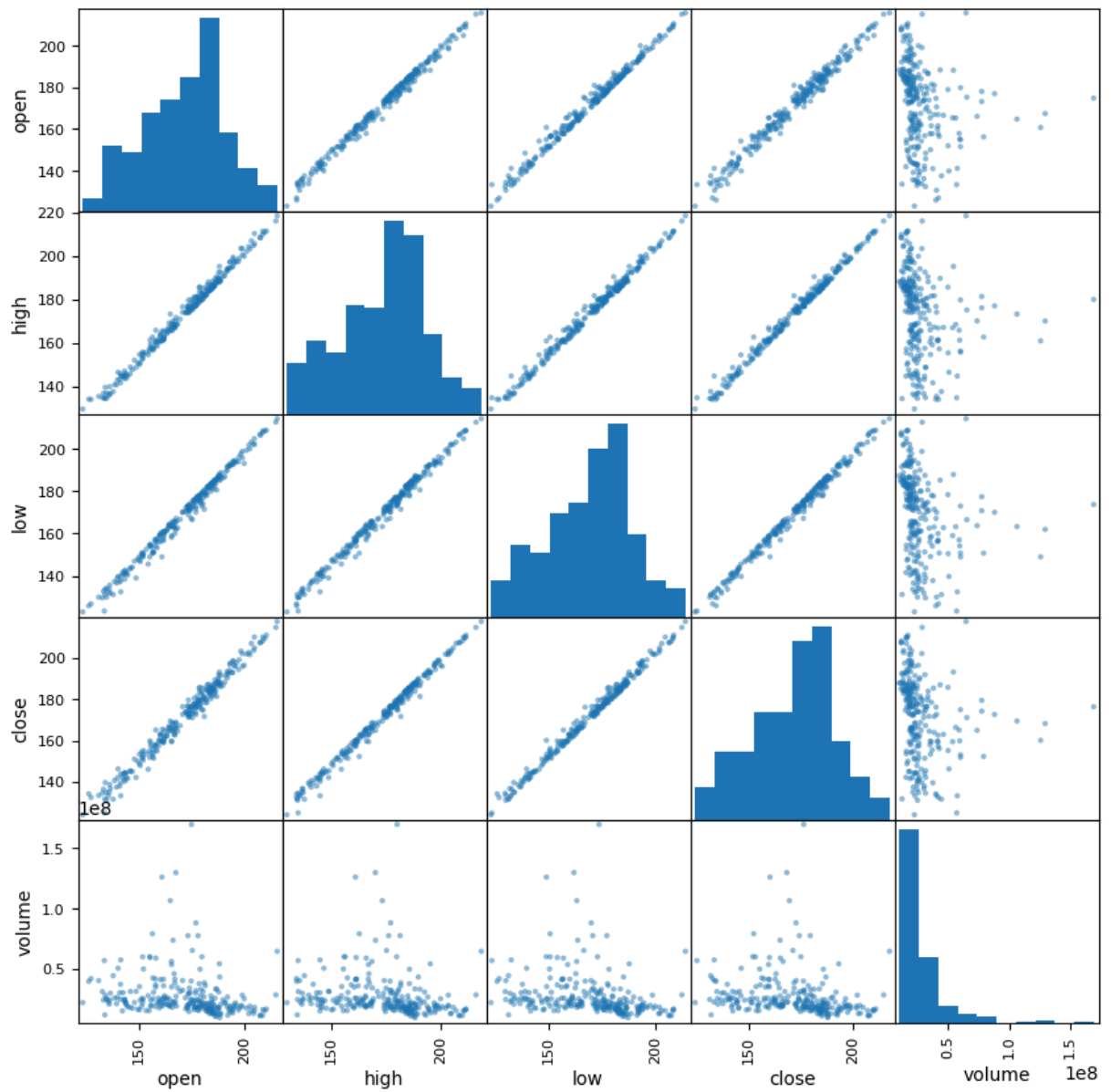# Setup

```python
In [ ]:  %matplotlib inline
         import matplotlib.pyplot as plt
         import numpy as np
         import pandas as pd
         fb = pd.read_csv(
         '/content/fb_stock_prices_2018.csv', index_col='date', parse_dates=True
         )
```

Scatter matrix

```python
In [ ]:  from pandas.plotting import scatter_matrix
         scatter_matrix(fb, figsize=(10, 10))
```

```
Out[ ]:  array([[<Axes: xlabel='open', ylabel='open'>,
                  <Axes: xlabel='high', ylabel='open'>,
                  <Axes: xlabel='low', ylabel='open'>,
                  <Axes: xlabel='close', ylabel='open'>,
                  <Axes: xlabel='volume', ylabel='open'>],
                 [<Axes: xlabel='open', ylabel='high'>,
                  <Axes: xlabel='high', ylabel='high'>,
                  <Axes: xlabel='low', ylabel='high'>,
                  <Axes: xlabel='close', ylabel='high'>,
                  <Axes: xlabel='volume', ylabel='high'>],
                 [<Axes: xlabel='open', ylabel='low'>,
                  <Axes: xlabel='high', ylabel='low'>,
                  <Axes: xlabel='low', ylabel='low'>,
                  <Axes: xlabel='close', ylabel='low'>,
                  <Axes: xlabel='volume', ylabel='low'>],
                 [<Axes: xlabel='open', ylabel='close'>,
                  <Axes: xlabel='high', ylabel='close'>,
                  <Axes: xlabel='low', ylabel='close'>,
                  <Axes: xlabel='close', ylabel='close'>,
                  <Axes: xlabel='volume', ylabel='close'>],
                 [<Axes: xlabel='open', ylabel='volume'>,
                  <Axes: xlabel='high', ylabel='volume'>,
                  <Axes: xlabel='low', ylabel='volume'>,
                  <Axes: xlabel='close', ylabel='volume'>,
                  <Axes: xlabel='volume', ylabel='volume'>]], dtype=object)
```

Changing the diagonal from histograms to KDE:

```
In [ ]: scatter_matrix(fb, figsize=(10, 10), diagonal='kde')
```

```
Out[ ]:  array([[<Axes: xlabel='open', ylabel='open'>,
                  <Axes: xlabel='high', ylabel='open'>,
                  <Axes: xlabel='low', ylabel='open'>,
                  <Axes: xlabel='close', ylabel='open'>,
                  <Axes: xlabel='volume', ylabel='open'>],
                 [<Axes: xlabel='open', ylabel='high'>,
                  <Axes: xlabel='high', ylabel='high'>,
                  <Axes: xlabel='low', ylabel='high'>,
                  <Axes: xlabel='close', ylabel='high'>,
                  <Axes: xlabel='volume', ylabel='high'>],
                 [<Axes: xlabel='open', ylabel='low'>,
                  <Axes: xlabel='high', ylabel='low'>,
                  <Axes: xlabel='low', ylabel='low'>,
                  <Axes: xlabel='close', ylabel='low'>,
                  <Axes: xlabel='volume', ylabel='low'>],
                 [<Axes: xlabel='open', ylabel='close'>,
                  <Axes: xlabel='high', ylabel='close'>,
                  <Axes: xlabel='low', ylabel='close'>,
                  <Axes: xlabel='close', ylabel='close'>,
                  <Axes: xlabel='volume', ylabel='close'>],
                 [<Axes: xlabel='open', ylabel='volume'>,
                  <Axes: xlabel='high', ylabel='volume'>,
                  <Axes: xlabel='low', ylabel='volume'>,
                  <Axes: xlabel='close', ylabel='volume'>,
                  <Axes: xlabel='volume', ylabel='volume'>]], dtype=object)
```
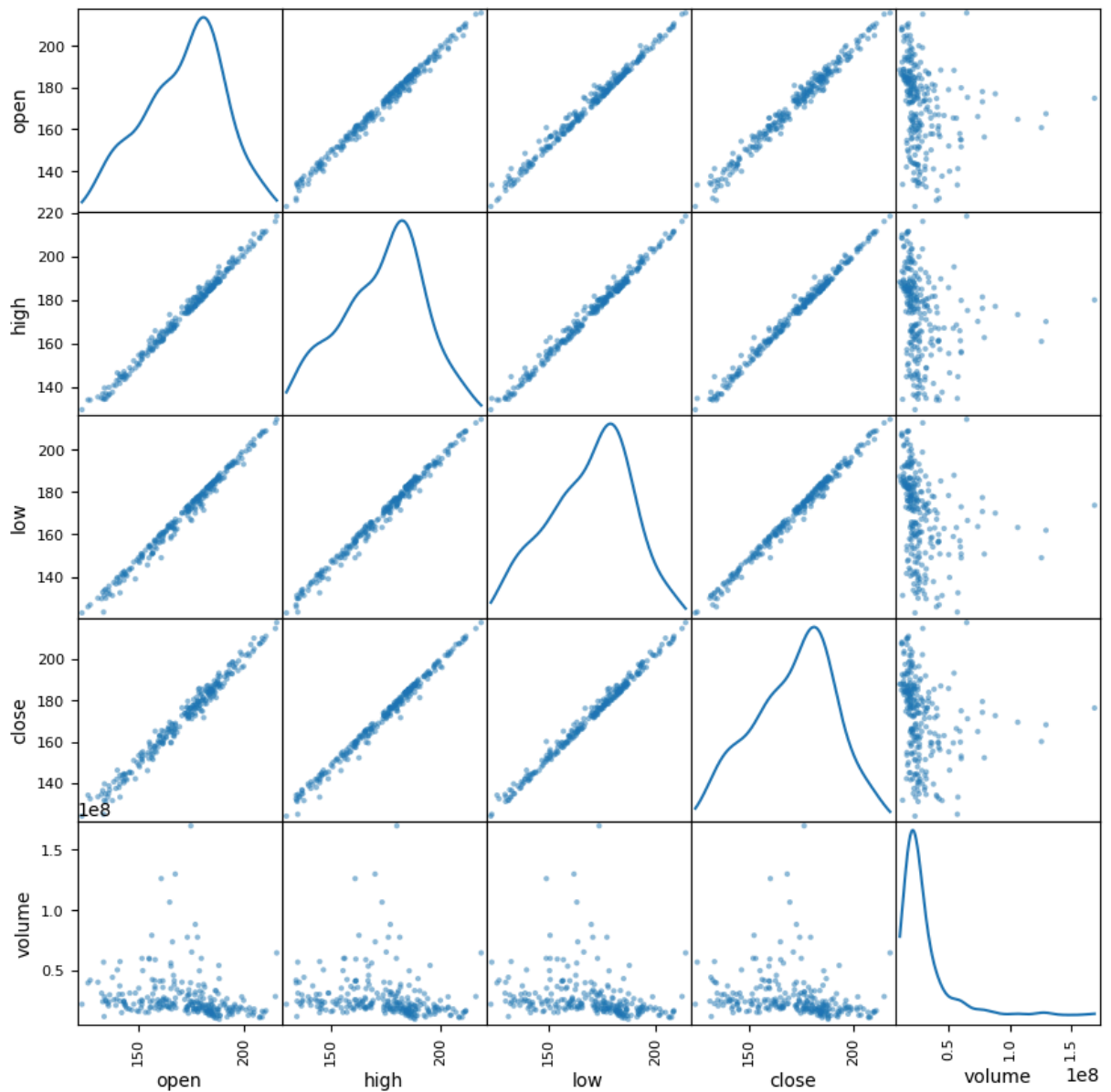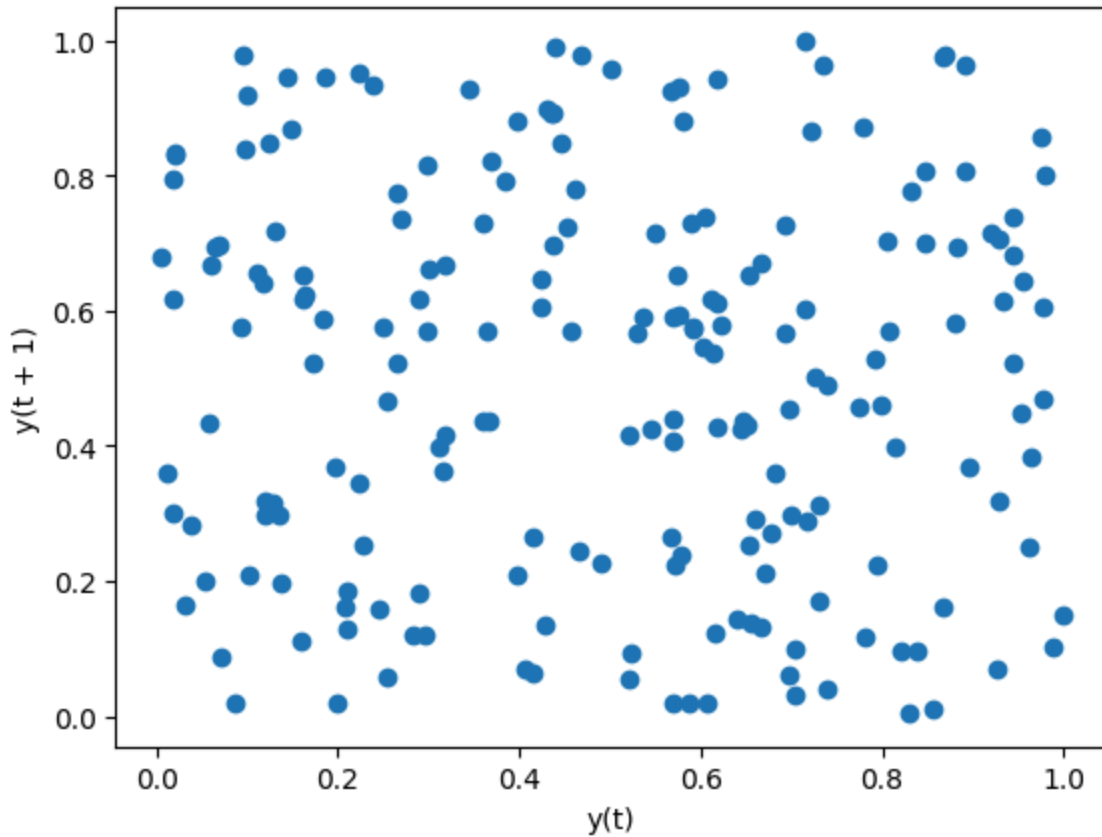
Lag plot

Lag plots let us see how the variable correlations with past observations of itself. Random data has no pattern:

```
In [ ]:  from pandas.plotting import lag_plot
         np.random.seed(0) # make this repeatable
         lag_plot(pd.Series(np.random.random(size=200)))

Out[ ]:  <Axes: xlabel='y(t)', ylabel='y(t + 1)'>
```
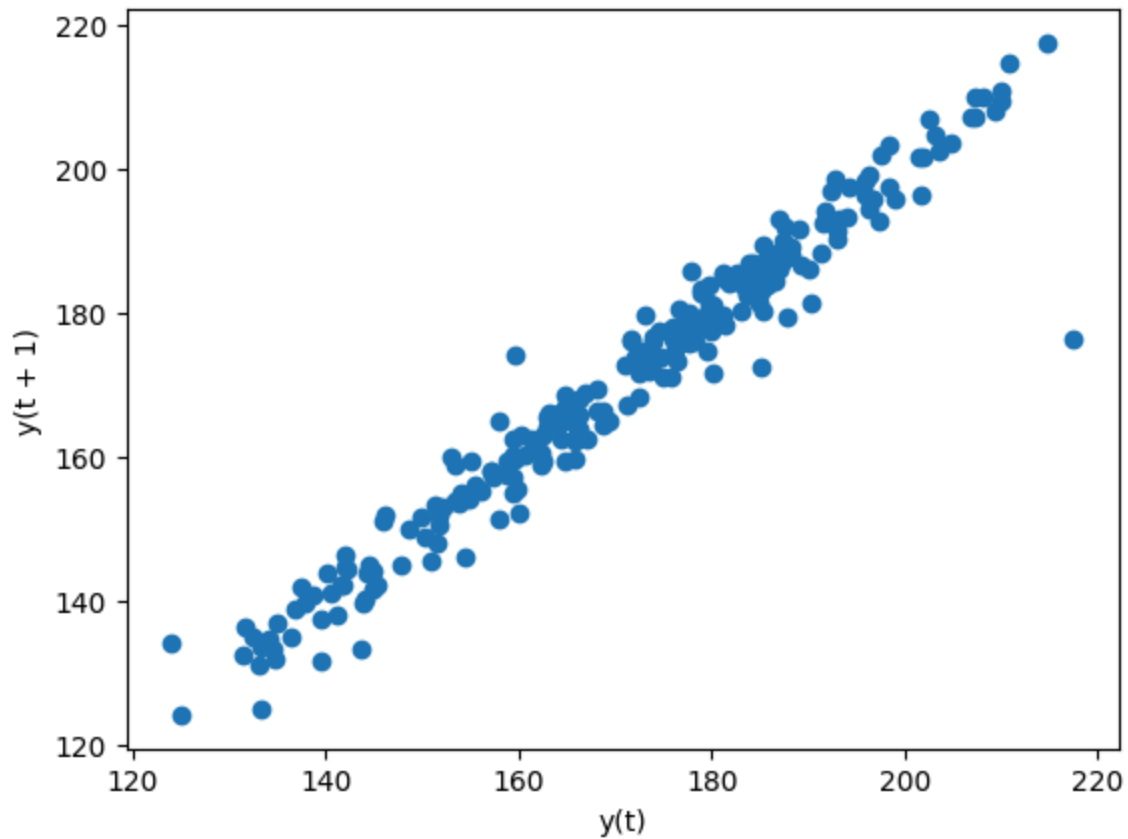
Data with some level of correlation to itself (autocorrelation) may have patterns. Stock prices are highly auto-correlated:

In [ ]:
```
lag_plot(fb.close)
```
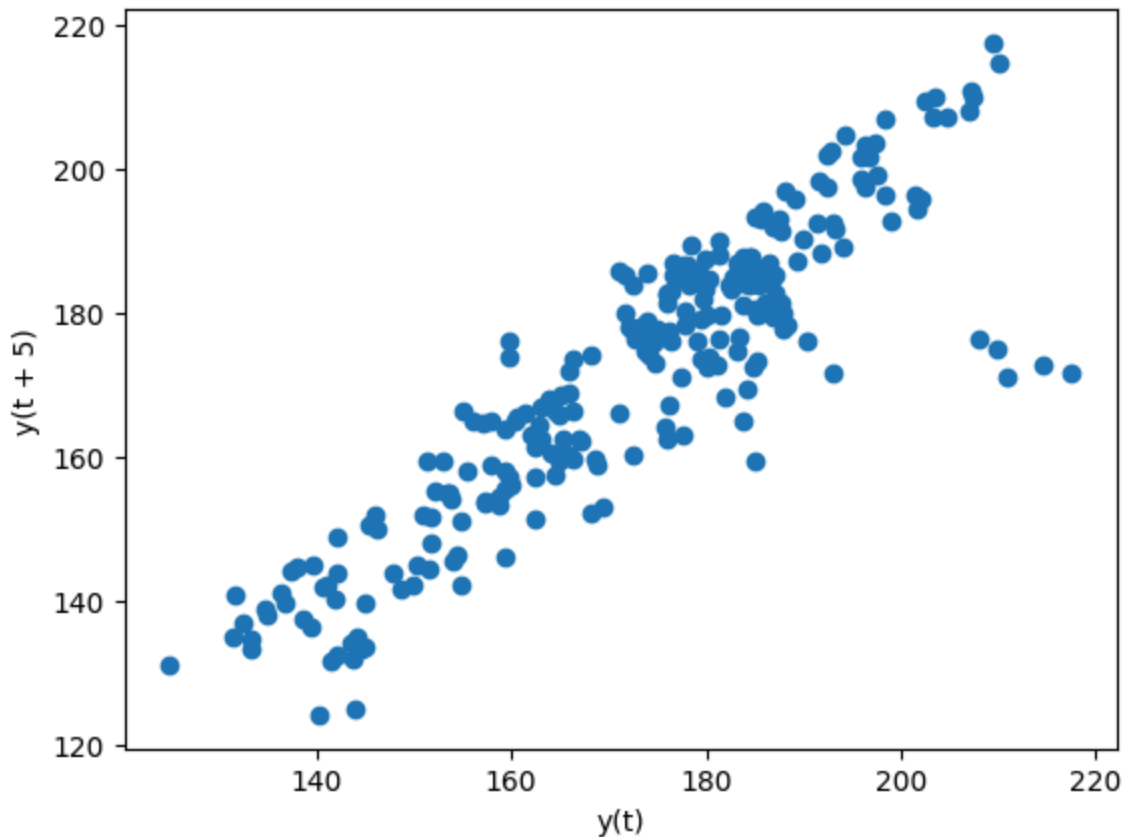
Out[ ]:   `<Axes: xlabel='y(t)', ylabel='y(t + 1)'>`

The default lag is 1, but we can alter this with the lag parameter. Let's look at a 5 day lag (a week of trading activity):

```
In [ ]:  lag_plot(fb.close, lag=5)
```

```
Out[ ]: <Axes: xlabel='y(t)', ylabel='y(t + 5)'>
```

# Autocorrelation plots

We can use the autocorrelation plot to see if this relationship may be meaningful or just noise. Random data will not have any significant autocorrelation (it stays within the bounds below):

```
In [ ]:  from pandas.plotting import autocorrelation_plot
         np.random.seed(0) # make this repeatable
         autocorrelation_plot(pd.Series(np.random.random(size=200)))
```

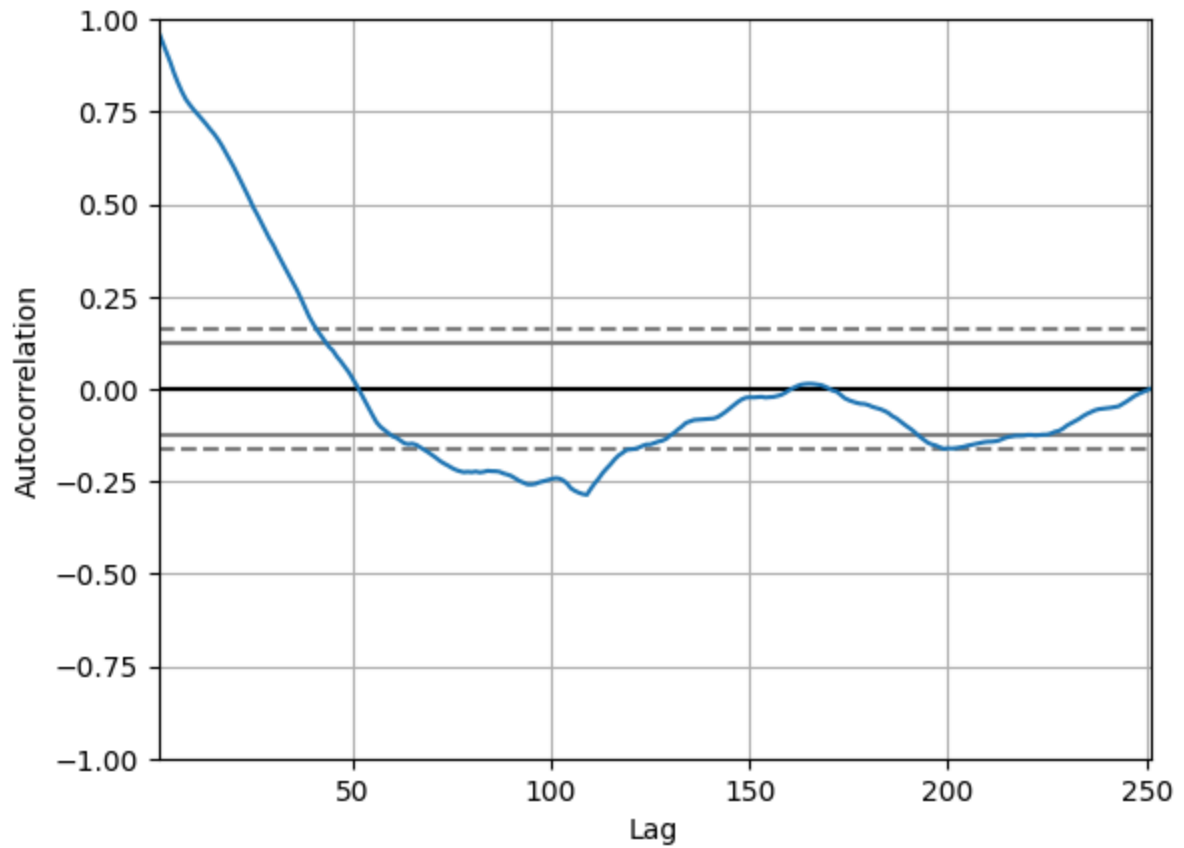```
Out[ ]: <Axes: xlabel='Lag', ylabel='Autocorrelation'>
```
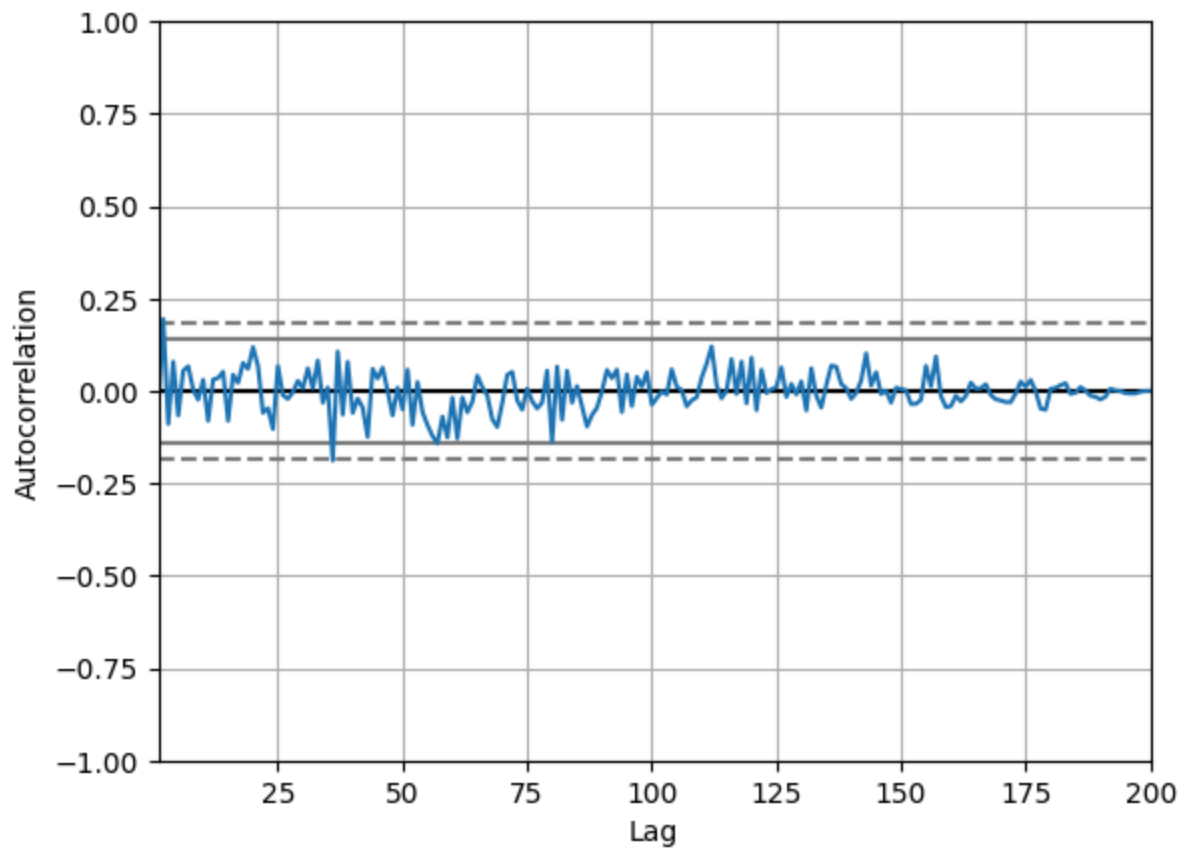
Stock data, on the other hand, does have significant autocorrelation:

```
In [ ]: autocorrelation_plot(fb.close)
```

```
Out[ ]: <Axes: xlabel='Lag', ylabel='Autocorrelation'>
```

Bootstrap plot

This plot helps us understsand the uncertainty in out summary statistics:

```
In [ ]:  from pandas.plotting import bootstrap_plot
         fig = bootstrap_plot(fb.volume, fig=plt.figure(figsize=(10, 6)))
```