4/2/24, 10:32 PM Hands-on Activity 9.6 Customizing Visualizations.ipynb - Colaboratory

volume 1e8

close

low

pandas.plotting subpackage

```
Setup
```

```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
fb = pd.read_csv(
  'fb_stock_prices_2018.csv', index_col='date', parse_dates=True
```

from pandas.plotting import scatter_matrix # scatter matrix
scatter_matrix(fb, figsize=(10, 10))

```
⇒ 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)
      Ö 160 -
        140 -
        220
        200 -
     dgid 180 -
        200 -
        180 -
        200 -
        1.5
```

Changing the diagonal from histograms to KDE:

open

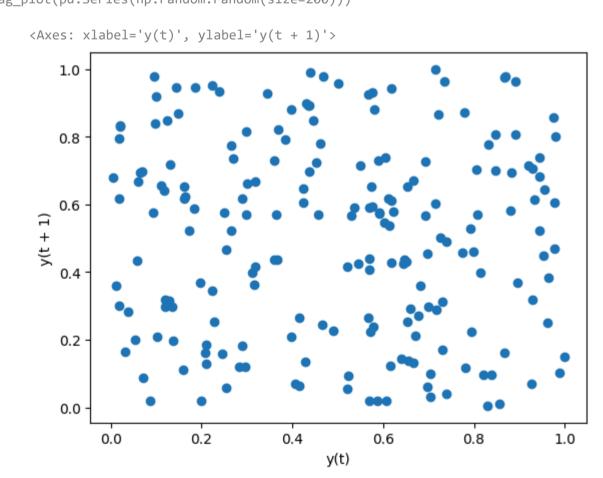
scatter_matrix(fb, figsize=(10, 10), diagonal='kde') # change the diagonal section to kde

```
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)
  _ 180 -
   140
   220
 dgi
180
   200 -
   180
```

Lag plot

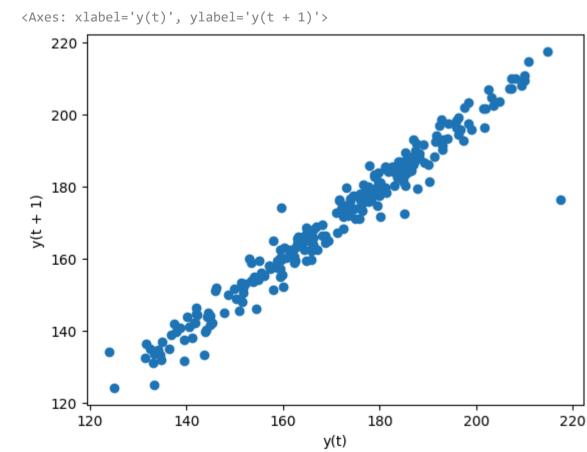
Lag plots let us see how the variable correlations with past observations of itself. Random data has no pattern:

from pandas.plotting import lag_plot # lag plot
np.random.seed(0) # make this repeatable
lag_plot(pd.Series(np.random.random(size=200)))



Data with some level of correlation to itself (autocorrelation) may have patterns. Stock prices are highly auto-correlated:

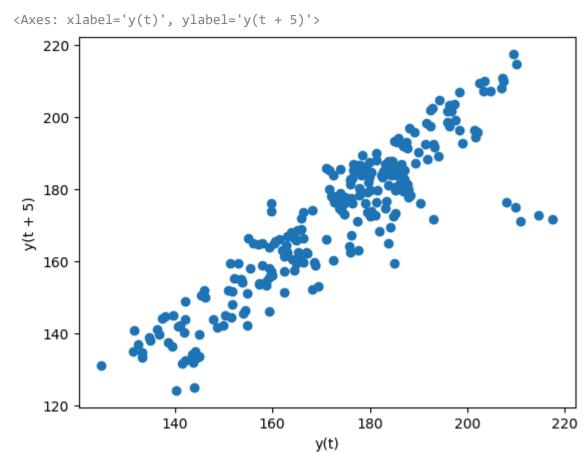
lag_plot(fb.close) # lag plotting closing price shows the correlation between the plots



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

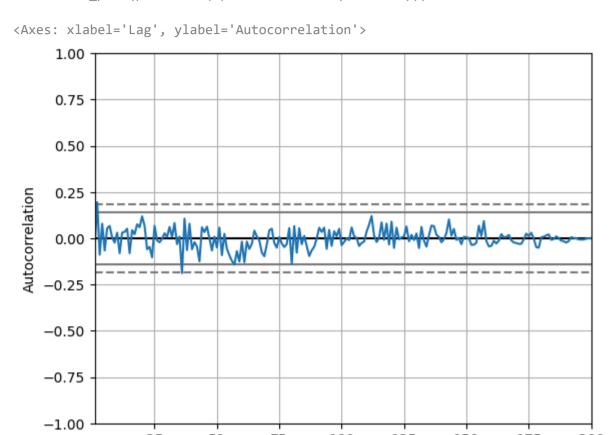
lag_plot(fb.close, lag=5) # 5 day lag

4/2/24, 10:32 PM Hands-on Activity 9.6 Customizing Visualizations.ipynb - Colaboratory



Autocorrelation plots

from pandas.plotting import autocorrelation_plot # autocorrelation plot
np.random.seed(0) # make this repeatable
autocorrelation_plot(pd.Series(np.random.random(size=200)))



100 Lag

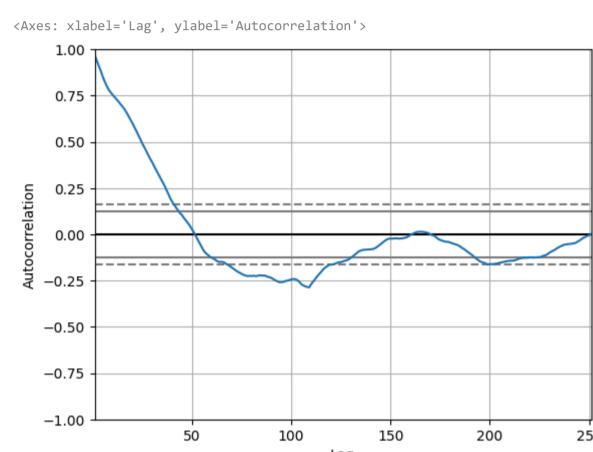
75

125 150 175 200

Stock data, on the other hand, does have significant autocorrelation:

25 50

autocorrelation_plot(fb.close)



Bootstrap plot

This plot helps us understand the uncertainty in our summary statistics

from pandas.plotting import bootstrap_plot # bootstrap_plot that helps understand uncertainty in summary statistics fig = bootstrap_plot(fb.volume, fig=plt.figure(figsize=(10, 6)))

