Formatting Plots

Marquez, Keith Leigh Zhen R.

About the Data In this notebook, we will be working with Facebook's stock price throughout 2018 (obtained using the stock_analysis package).

Setup

```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns

fb = pd.read_csv(
    '/content/fb_stock_prices_2018.csv', index_col='date', parse_dates=True
)
```

+ Code + Text

Titles and Axis Labels

plt.suptitle() adds a title to plots and subplots

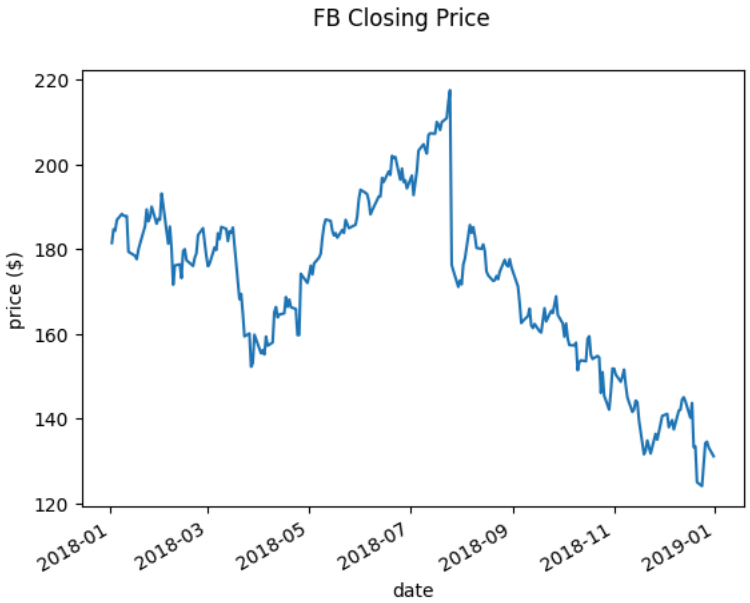
plt.title() adds a title to a single plot. Note if you use subplots, it will only put the title on the last subplot, so you will need to use

plt.xlabel() labels the x-axis

plt.ylabel() labels the y-axis

```
fb.close.plot()
plt.suptitle('FB Closing Price')
plt.xlabel('date')
plt.ylabel('price ($)')
```

```
Text(0, 0.5, 'price ($)')
```

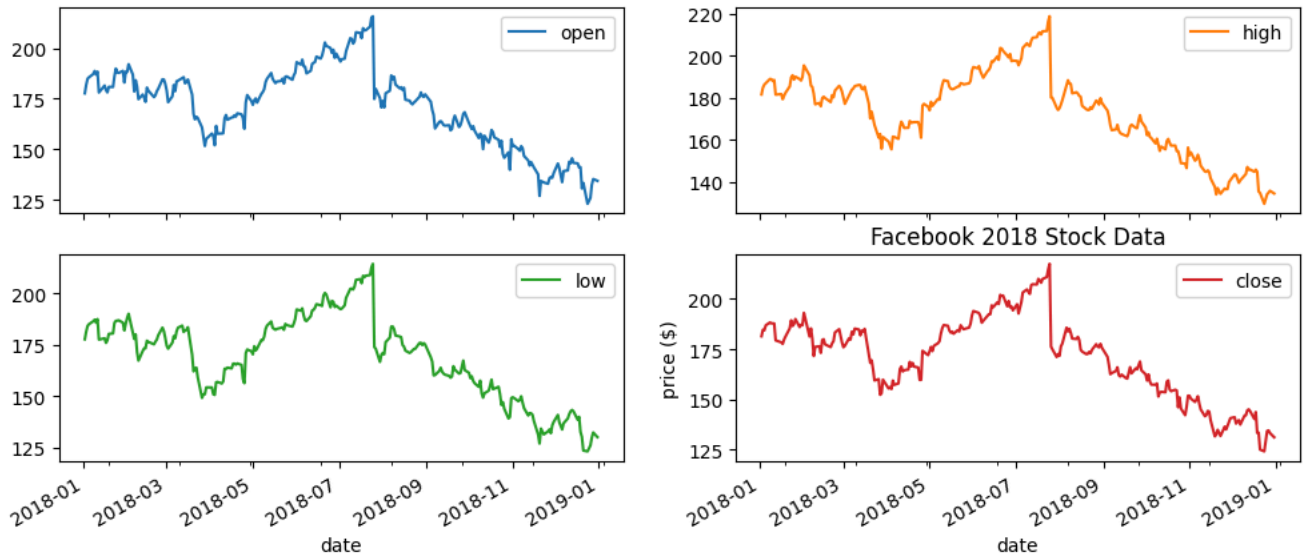


plt.suptitle() vs. plt.title()

Check out what happens when we call plt.title() with subplots:

```
# Plot the first four columns of fb DataFrame as subplots in a 2x2 layout
fb.iloc[:,:4].plot(subplots=True, layout=(2, 2), figsize=(12, 5))
# Set the title, xlabel, and ylabel for the plot
plt.title('Facebook 2018 Stock Data') #plt.title()
plt.xlabel('date')
plt.ylabel('price ($)')
```

```
Text(0, 0.5, 'price ($)')
```

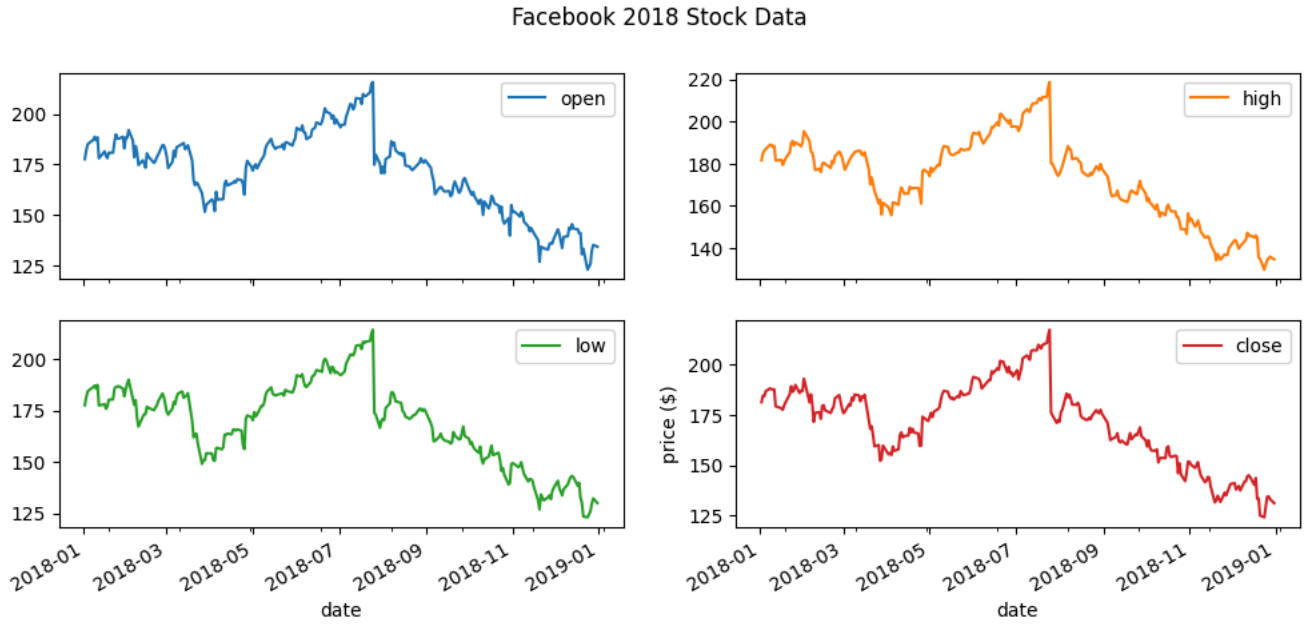


Simply getting into the habit of using plt.suptitle() instead of plt.title() will save you this confusion:

Double-click (or enter) to edit

```
# Plot the first four columns of fb DataFrame as subplots in a 2x2 layout
fb.iloc[:,:4].plot(subplots=True, layout=(2, 2), figsize=(12, 5))
# Set the title, xlabel, and ylabel for the plot
plt.suptitle('Facebook 2018 Stock Data') # plt.suptitle()
plt.xlabel('date')
plt.ylabel('price ($)')
```

```
Text(0, 0.5, 'price ($)')
```

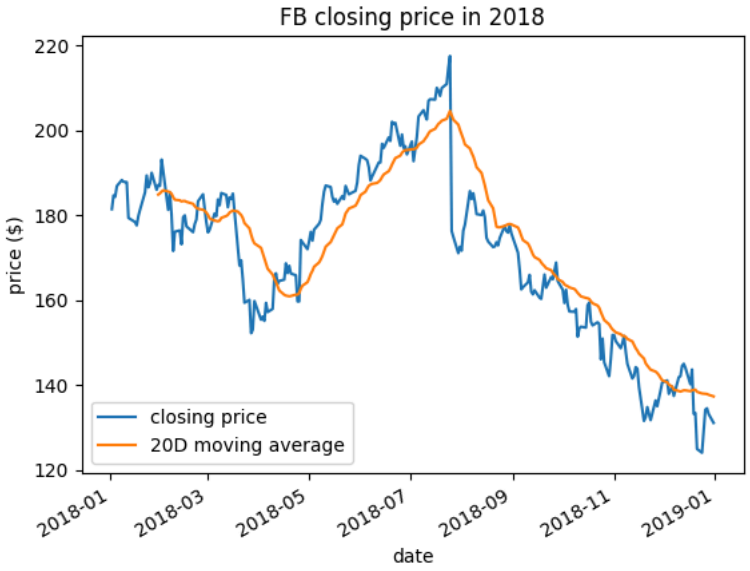


Legends

plt.legend() adds a legend to the plot. We can specify where to place it with the locparameter:

```
fb.assign( # Calculate the 20-day moving average and assign it to a new column 'ma'
    ma=lambda x: x.close.rolling(20).mean()
    ).plot(
        y=['close', 'ma'],
        title='FB closing price in 2018',
        label=['closing price', '20D moving average'] # Plot the closing price and the 20-day moving average
)
plt.legend(loc='lower left')
plt.ylabel('price ($)')
```

```
Text(0, 0.5, 'price ($)')
```



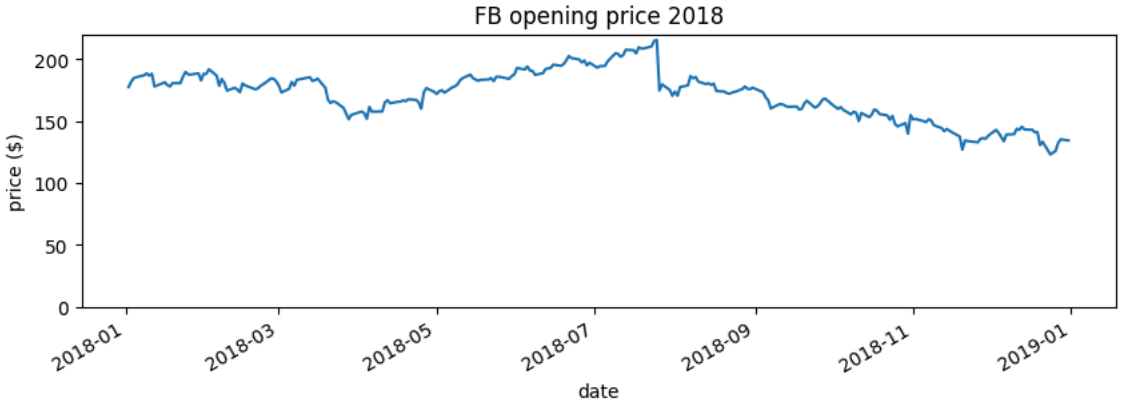
Formatting Axes

Specifying axis limits

plt.xlim() and plt.ylim() can be used to specify the minimum and maximum values for the axis. Passing None will have matplotlib determine the limit.

```
fb.open.plot(figsize=(10, 3), title='FB opening price 2018') # Plot the opening price of Facebook stock in 2018 with specified figure size an
plt.ylim(0, None) # Set the y-axis limit to start from 0
plt.ylabel('price ($)')
```

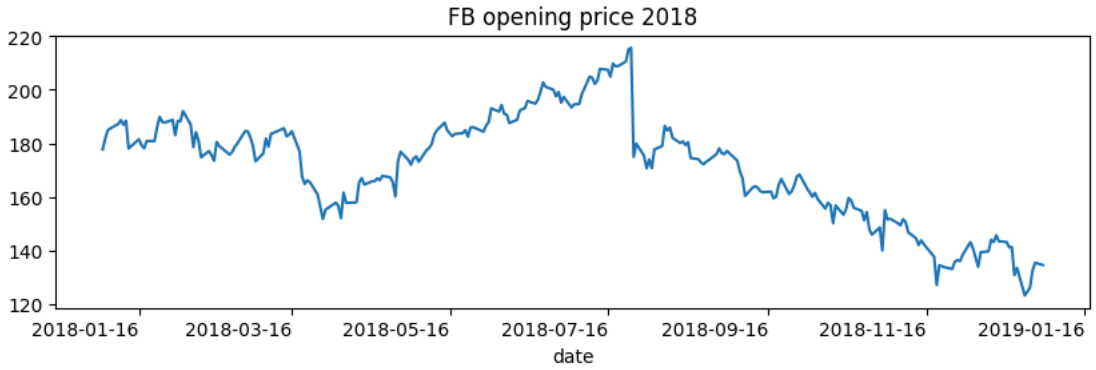
```
Text(0, 0.5, 'price ($)')
```



Formatting the Axis Ticks We can use plt.xticks() and plt.yticks() to provide tick labels and specify, which ticks to show. Here, we show every other month:

```
import calendar
fb.open.plot(figsize=(10, 3), rot=0, title='FB opening price 2018')
locs, labels = plt.xticks()
plt.xticks(locs + 15, calendar.month_name[1::2])
plt.ylabel('price ($)')
```

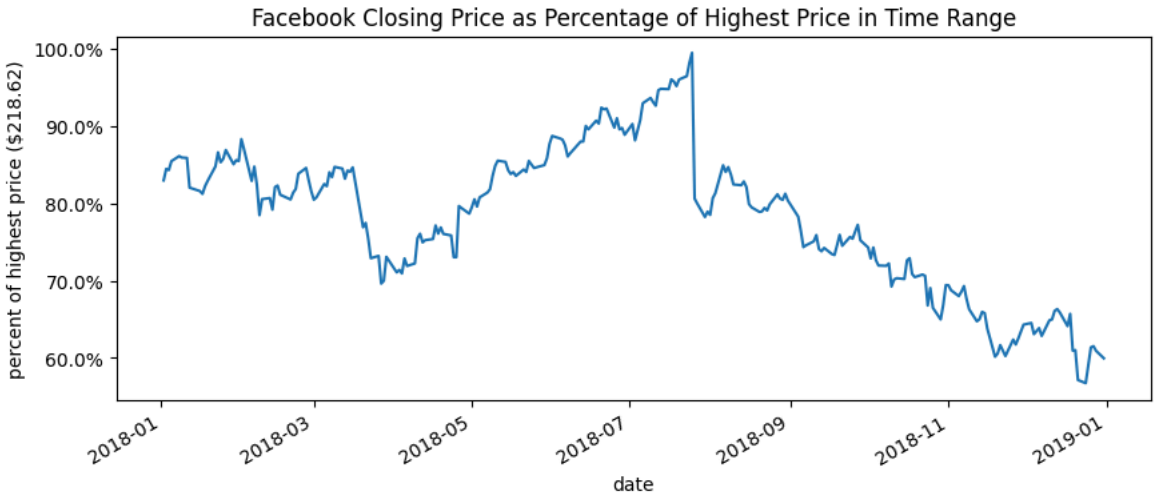
```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-11-efbfb8018ae1> in <cell line: 4>()
      2 fb.open.plot(figsize=(10, 3), rot=0, title='FB opening price 2018')
      3 locs, labels = plt.xticks()
----> 4 plt.xticks(locs + 15, calendar.month_name[1::2])
      5 plt.ylabel('price ($)')

                                 3 frames
/usr/local/lib/python3.10/dist-packages/matplotlib/axis.py in set_ticklabels(self, labels, minor, fontdict, **kwargs)
   1967             # remove all tick labels, so only error for > 0 labels
   1968             if len(locator.locs) != len(labels) and len(labels) != 0:
-> 1969                 raise ValueError(
   1970                     "The number of FixedLocator locations"
   1971                     f" ({len(locator.locs)}), usually from a call to"

ValueError: The number of FixedLocator locations (7), usually from a call to set_ticks, does not match the number of labels (6).
```



Using ticker PercentFormatter We can use ticker.PercentFormatter and specify the denominator (set_major_formatter() method of the xaxis or xmax) to use when calculating the percentages. This gets passed to the yaxis on the Axes .

```
import matplotlib.ticker as ticker
ax = fb.close.plot(
    figsize=(10, 4),
    title='Facebook Closing Price as Percentage of Highest Price in Time Range'
)
ax.yaxis.set_major_formatter(
    ticker.PercentFormatter(xmax=fb.high.max())
)
ax.set_yticks([
    fb.high.max()*pct for pct in np.linspace(0.6, 1, num=5)
]) # show round percentages only (60%, 80%, etc.)
ax.set_ylabel(f'percent of highest price (${fb.high.max()})')
```

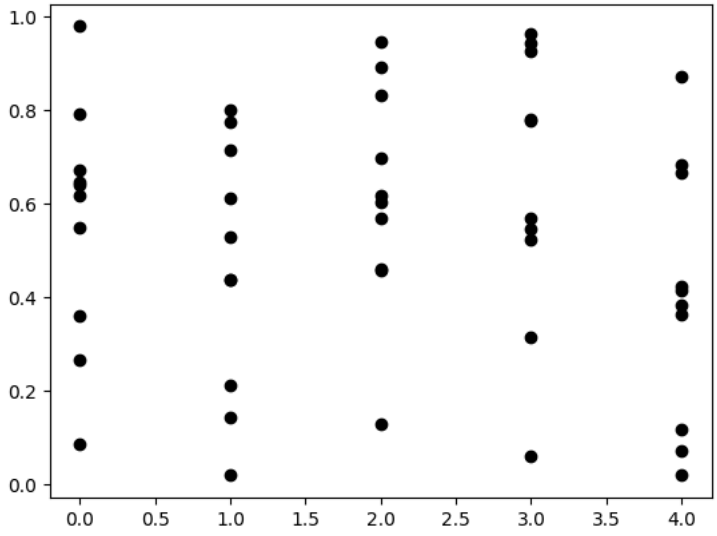
```
Text(0, 0.5, 'percent of highest price ($218.62)')
```



MultipleLocator Say we have the following data. The points only take on integer values for x

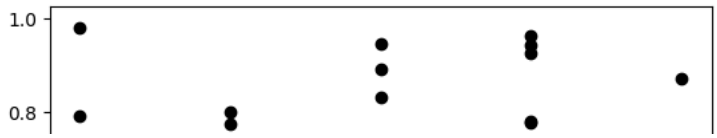
```
fig, ax = plt.subplots(1, 1)
np.random.seed(0) # Generate random data
ax.plot(np.tile(np.arange(0, 5), 10), np.random.rand(50), 'ko')
```

```
[<matplotlib.lines.Line2D at 0x7b7377007670>]
```



If we don't want to show decimal values on the x-axis, we can use the parameter. To get integer values, we use base=1:

```
fig, ax = plt.subplots(1, 1)
np.random.seed(0) # Generate random data
ax.plot(np.tile(np.arange(0, 5), 10), np.random.rand(50), 'ko')
ax.get_xaxis().set_major_locator(
    ticker.MultipleLocator(base=1) # Set major ticks on x-axis at intervals of 1
)
```


1.0
0.8