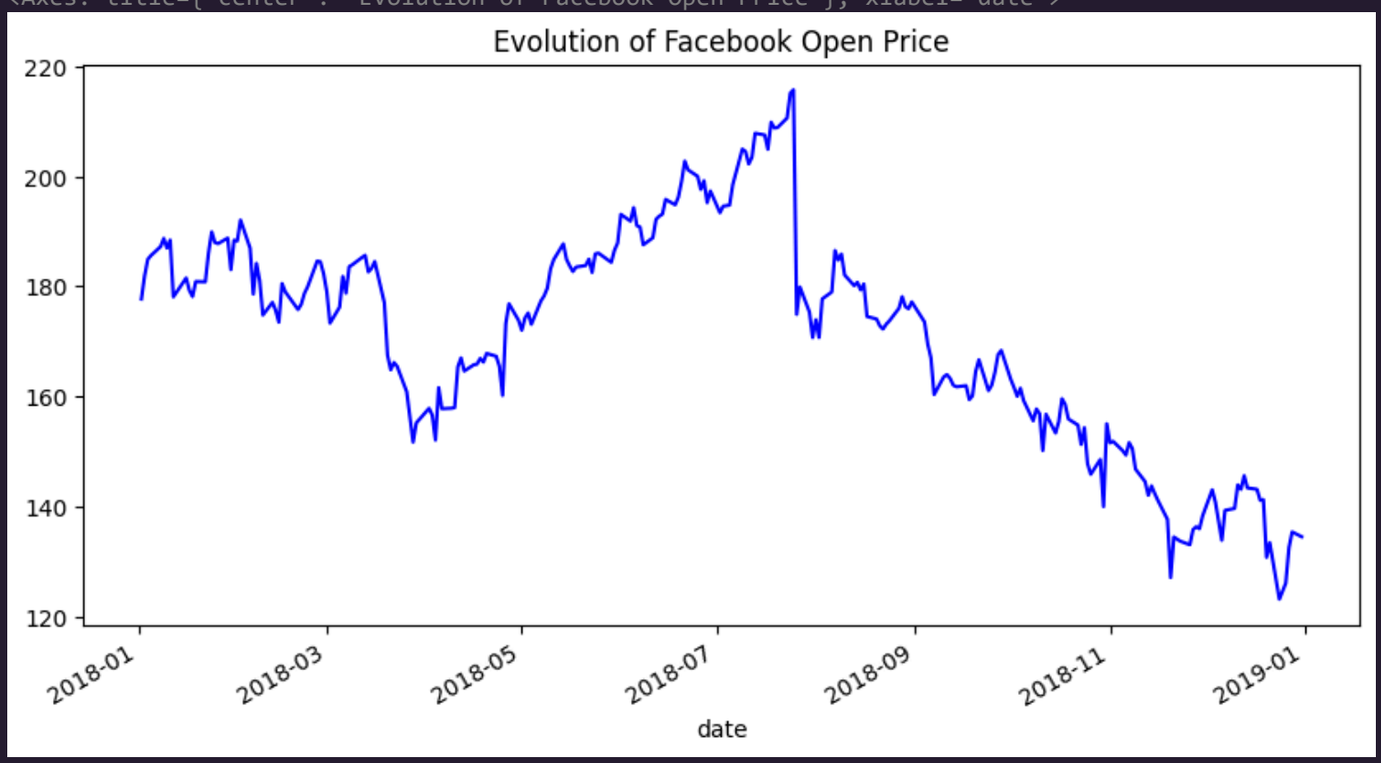
```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
fb = pd.read_csv(
 '/content/fb_stock_prices_2018.csv', index_col='date', parse_dates=True
)
quakes = pd.read_csv('/content/earthquakes-1.csv')
```

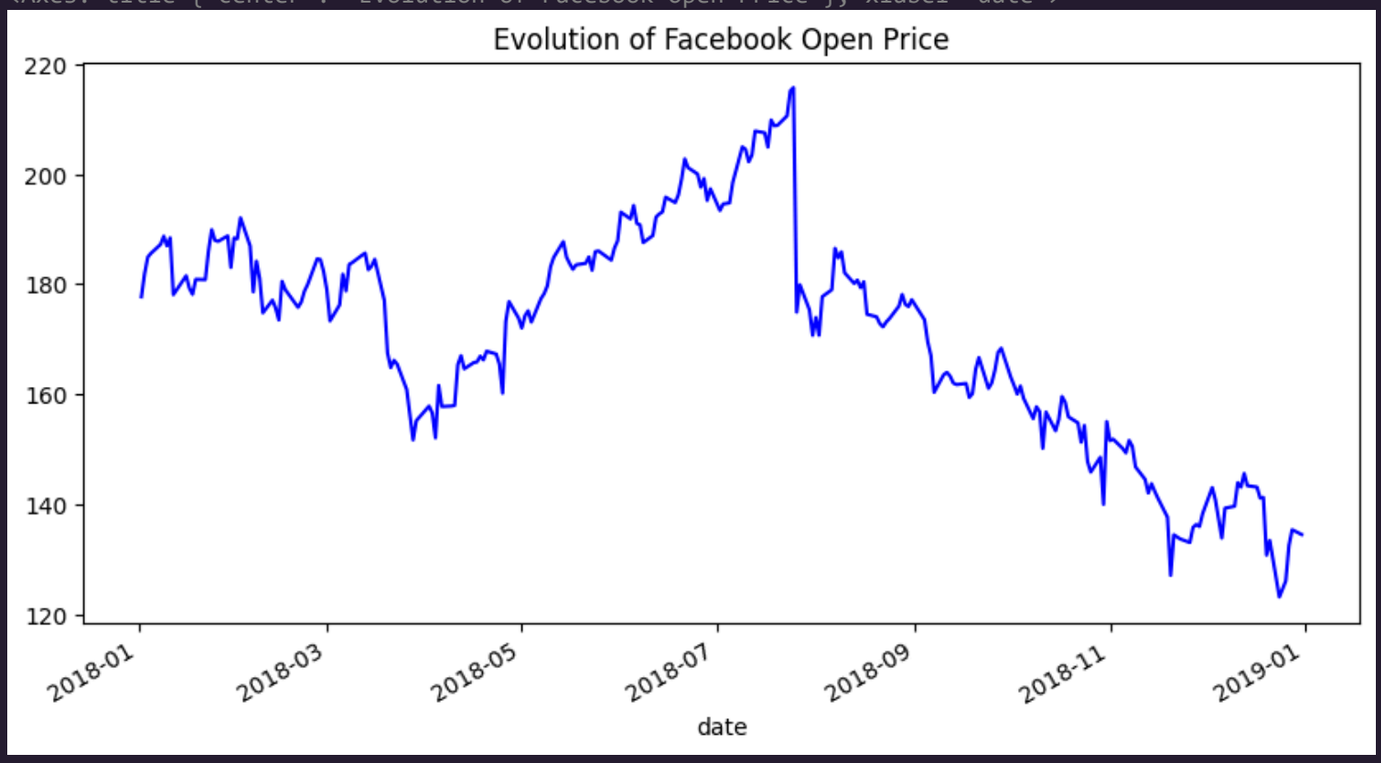
```
fb.plot(
 kind='line',
 y='open',
 figsize=(10, 5),
 style='b-',
 legend=False,
 title='Evolution of Facebook Open Price'
)

```

```
<Axes: title={'center': 'Evolution of Facebook Open Price'}, xlabel='date'>
```



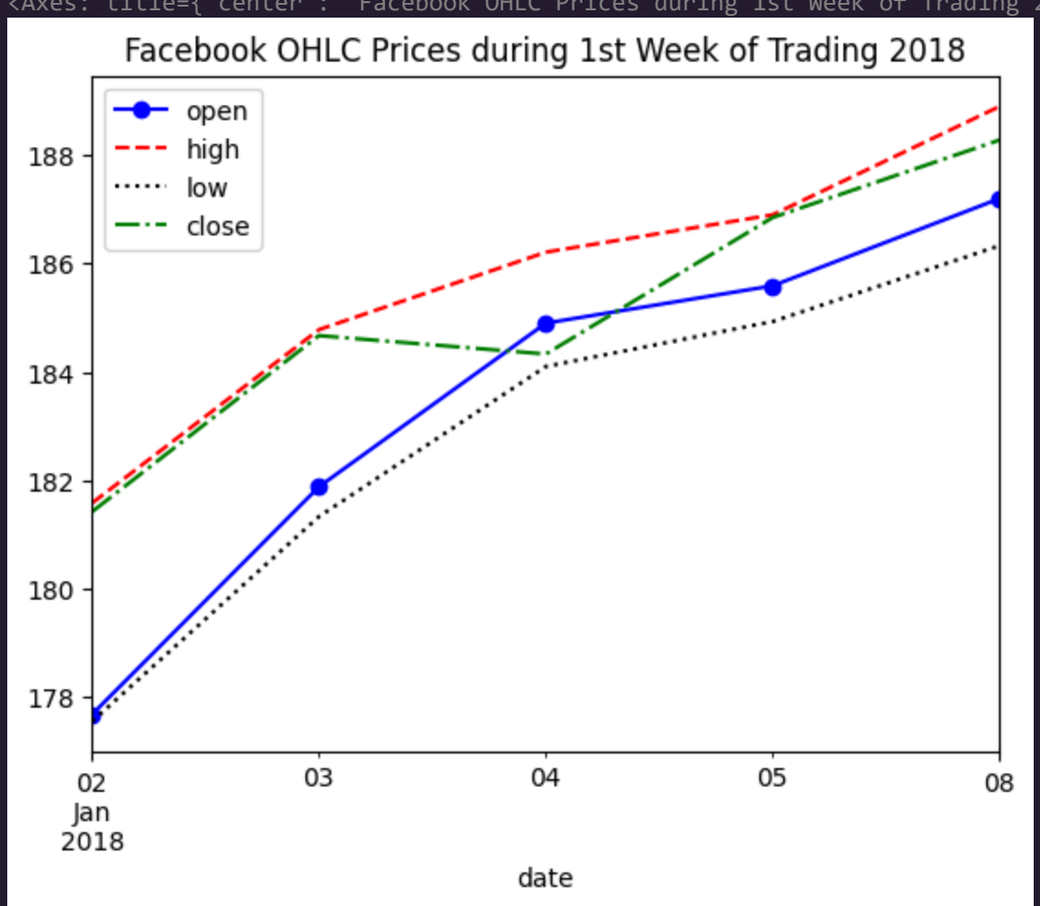

```
fb.plot(
 kind='line',
 y='open',
 figsize=(10, 5),
 color='blue',
 linestyle='solid',
 legend=False,
 title='Evolution of Facebook Open Price'
)

```

```
<Axes: title={'center': 'Evolution of Facebook Open Price'}, xlabel='date'>
```



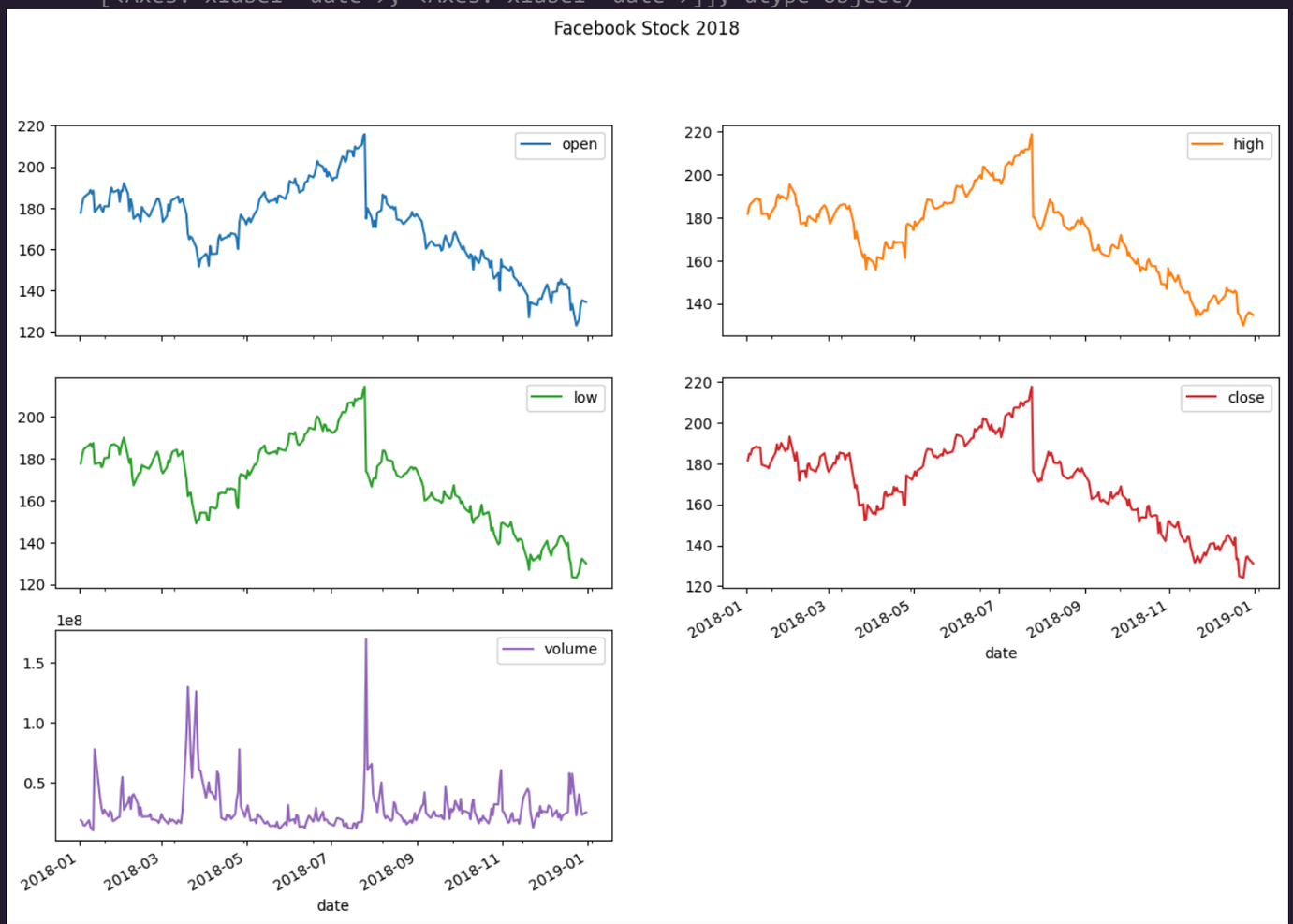

```
1 fb.iloc[:5,].plot(
2  y=['open', 'high', 'low', 'close'],
3  style=['b-o', 'r--', 'k:', 'g-.'],
4  title='Facebook OHLC Prices during 1st Week of Trading 2018'
5 )
```

```
<Axes: title={'center': 'Facebook OHLC Prices during 1st Week of Trading 2018'}, xlabel='date'>
```



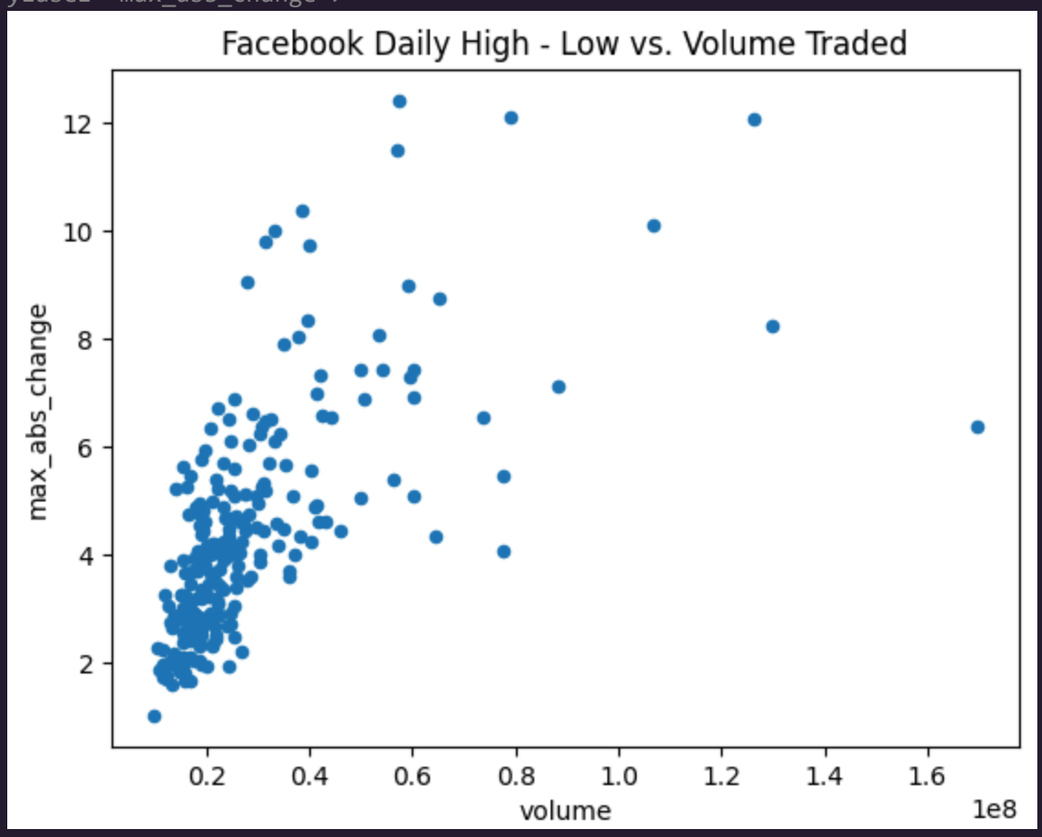

```
1 fb.plot(
2  kind='line',
3  subplots=True,
4  layout=(3,2),
5  figsize=(15,10),
6  title='Facebook Stock 2018'
7 )
8
```

```
array([[<Axes: xlabel='date'>, <Axes: xlabel='date'>],
       [<Axes: xlabel='date'>, <Axes: xlabel='date'>],
       [<Axes: xlabel='date'>, <Axes: xlabel='date'>]], dtype=object)
```



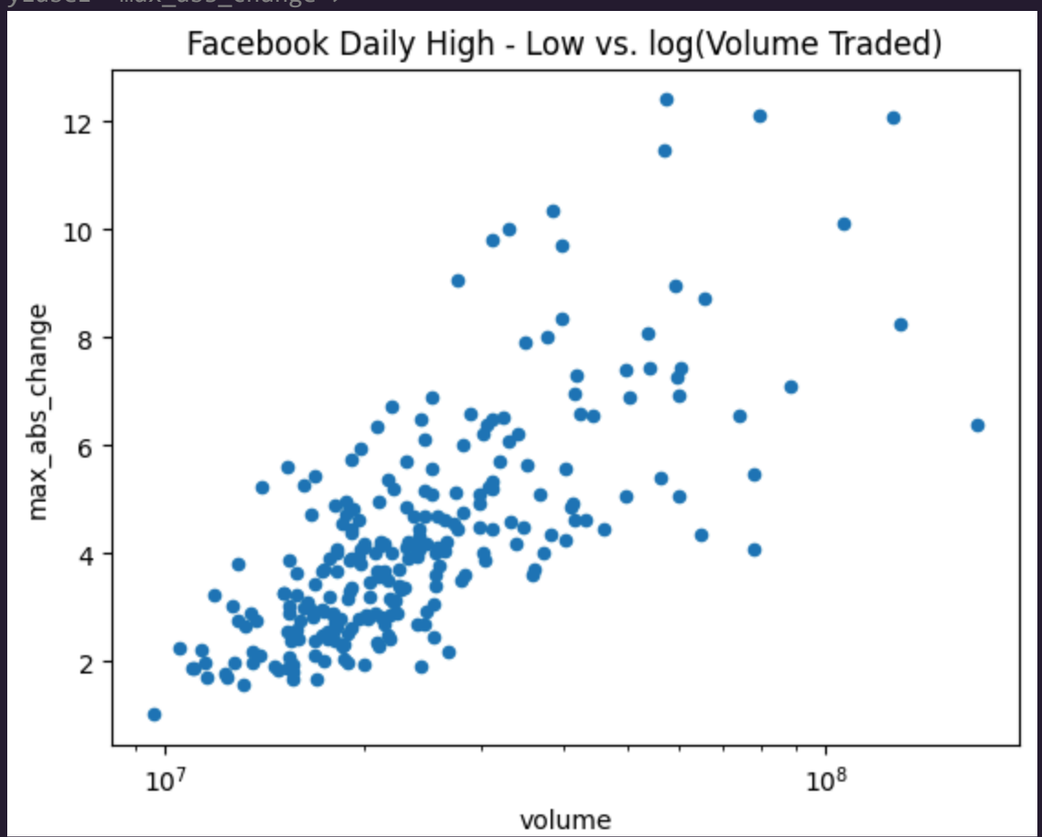

```
fb.assign(
 max_abs_change=fb.high - fb.low
).plot(
 kind='scatter', x='volume', y='max_abs_change',
 title='Facebook Daily High - Low vs. Volume Traded'
)

```

```
<Axes: title={'center': 'Facebook Daily High - Low vs. Volume Traded'}, xlabel='volume',
ylabel='max_abs_change'>
```



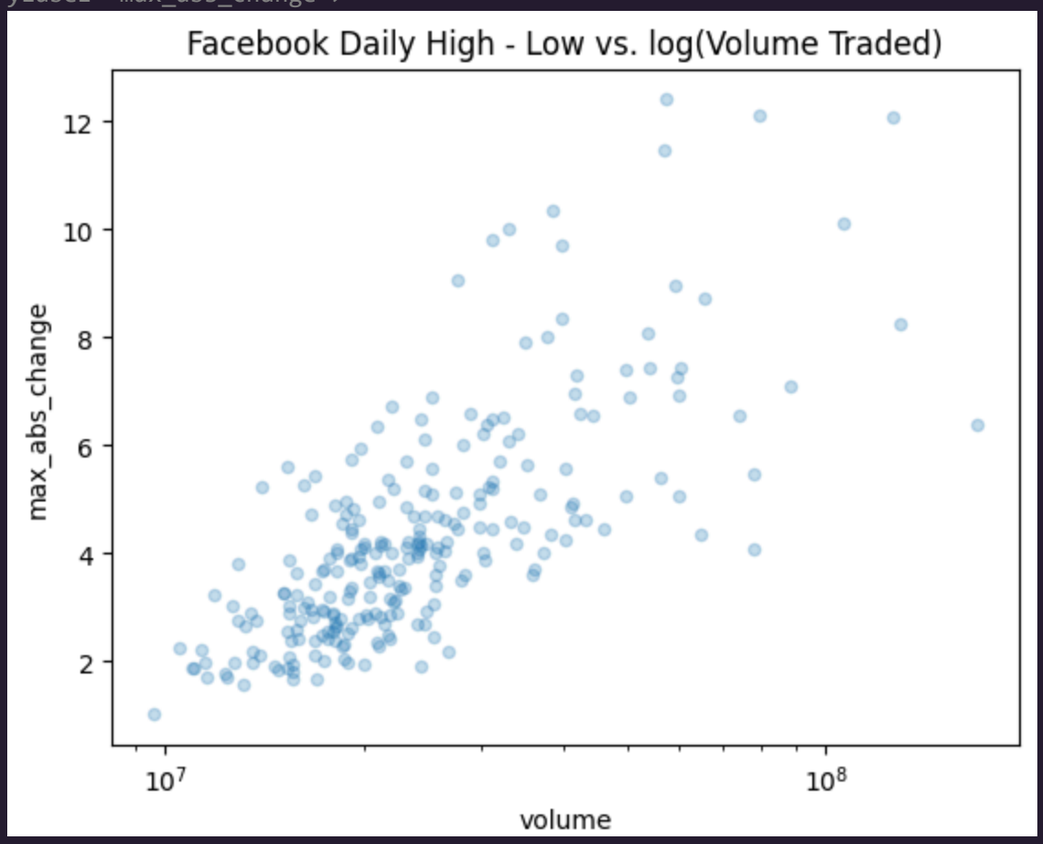

```
fb.assign(
 max_abs_change=fb.high - fb.low
).plot(
 kind='scatter', x='volume', y='max_abs_change',
 title='Facebook Daily High - Low vs. log(Volume Traded)',
 logx=True
)
```

```
<Axes: title={'center': 'Facebook Daily High - Low vs. log(Volume Traded)'}, xlabel='volume',
ylabel='max_abs_change'>
```



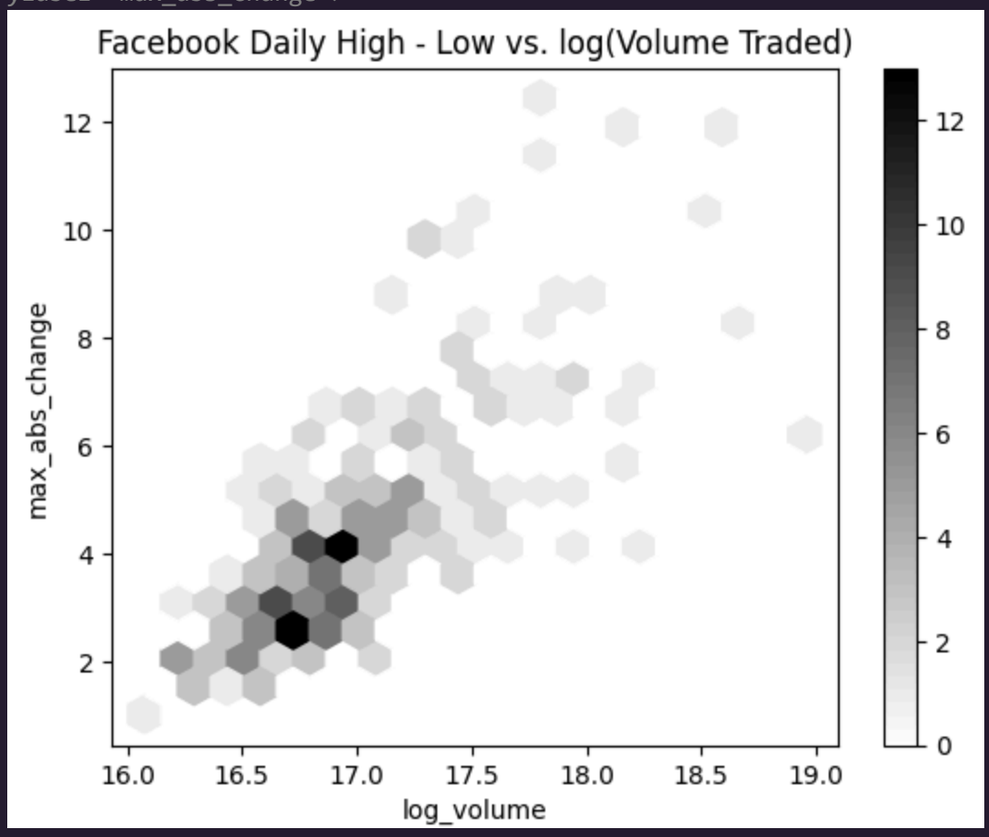

```
fb.assign(
 max_abs_change=fb.high - fb.low
).plot(
 kind='scatter', x='volume', y='max_abs_change',
 title='Facebook Daily High - Low vs. log(Volume Traded)',
 logx=True, alpha=0.25
)
```

```
<Axes: title={'center': 'Facebook Daily High - Low vs. log(Volume Traded)'}, xlabel='volume',
ylabel='max_abs_change'>
```



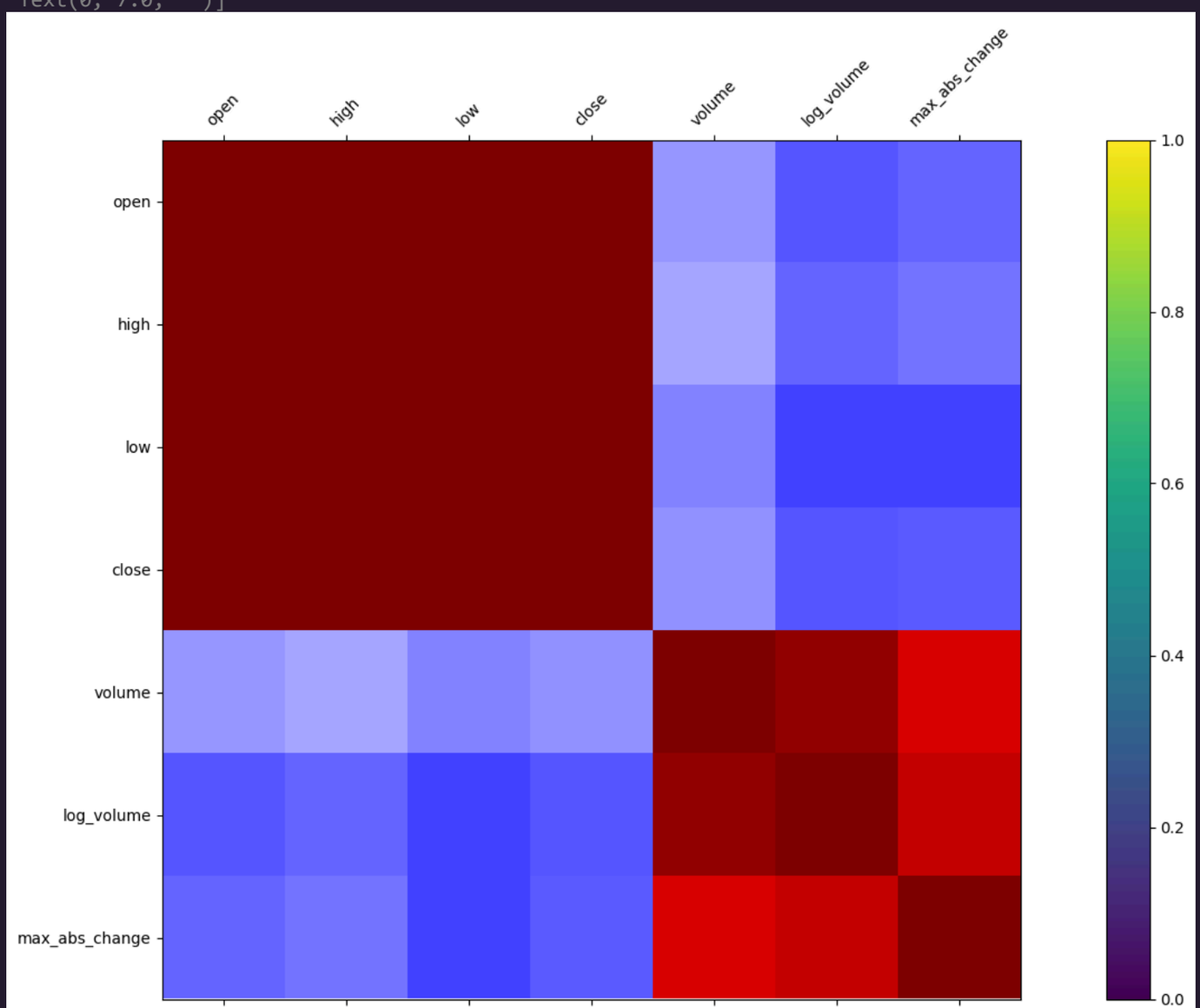

```
fb.assign(
 log_volume=np.log(fb.volume),
 max_abs_change=fb.high - fb.low
).plot(
 kind='hexbin',
 x='log_volume',
 y='max_abs_change',
 title='Facebook Daily High - Low vs. log(Volume Traded)',
 colormap='gray_r',
 gridsize=20,
 sharex=False # we have to pass this to see the x-axis due to a bug in this version of pandas
)

```

```
<Axes: title={'center': 'Facebook Daily High - Low vs. log(Volume Traded)'}, xlabel='log_volume',
ylabel='max_abs_change'>
```



```
fig, ax = plt.subplots(figsize=(20, 10))
fb_corr = fb.assign(
 log_volume=np.log(fb.volume),
 max_abs_change=fb.high - fb.low
).corr()
im = ax.matshow(fb_corr, cmap='seismic')
fig.colorbar(im.set_clim(-1, 1))
labels = [col.lower() for col in fb_corr.columns]
ax.set_xticklabels([''] + labels, rotation=45)
ax.set_yticklabels([''] + labels)
```

```
<ipython-input-15-d7b774946302>:7: MatplotlibDeprecationWarning: Unable to determine Axes to steal
  fig.colorbar(im.set_clim(-1, 1))
<ipython-input-15-d7b774946302>:9: UserWarning: FixedFormatter should only be used together with
  ax.set_xticklabels([''] + labels, rotation=45)
<ipython-input-15-d7b774946302>:10: UserWarning: FixedFormatter should only be used together with
  ax.set_yticklabels([''] + labels)
[Text(0, -1.0, ''),
 Text(0, 0.0, 'open'),
 Text(0, 1.0, 'high'),
 Text(0, 2.0, 'low'),
 Text(0, 3.0, 'close'),
 Text(0, 4.0, 'volume'),
 Text(0, 5.0, 'log_volume'),
 Text(0, 6.0, 'max_abs_change'),
 Text(0, 7.0, '')]
```

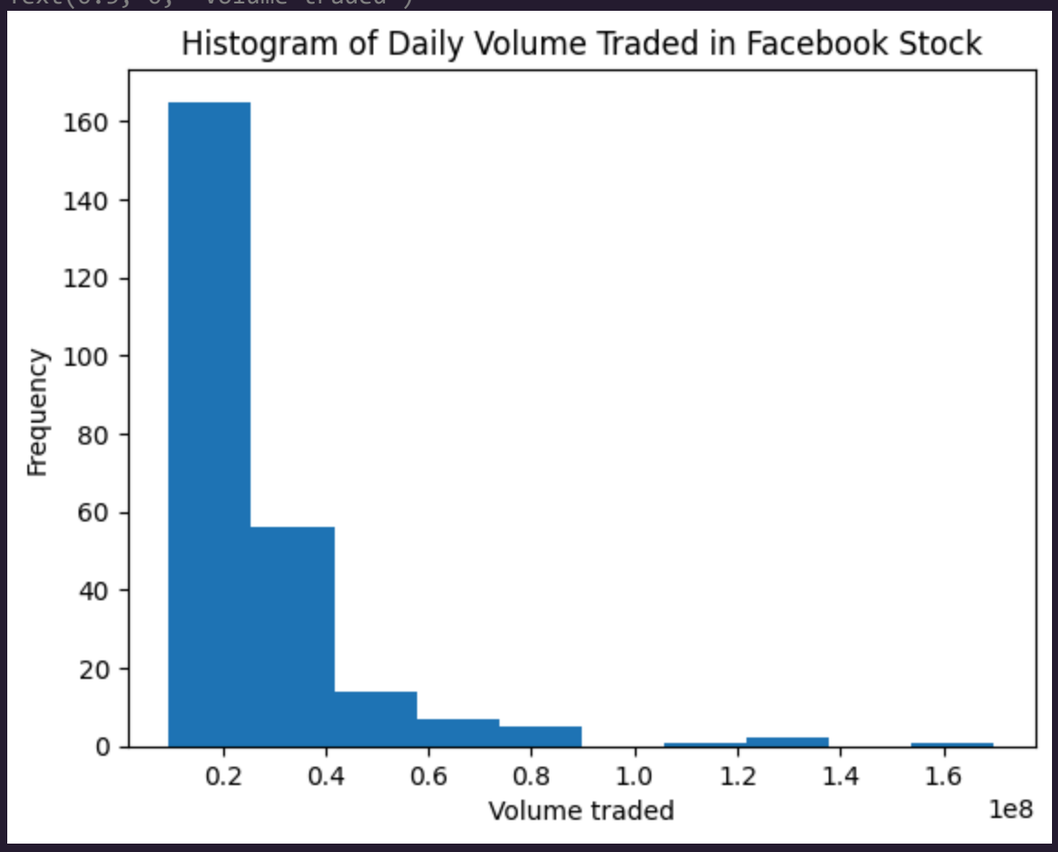



```
1 fb_corr.loc['max_abs_change', ['volume', 'log_volume']]
```

```
volume        0.642027
log_volume    0.731542
Name: max_abs_change, dtype: float64
```

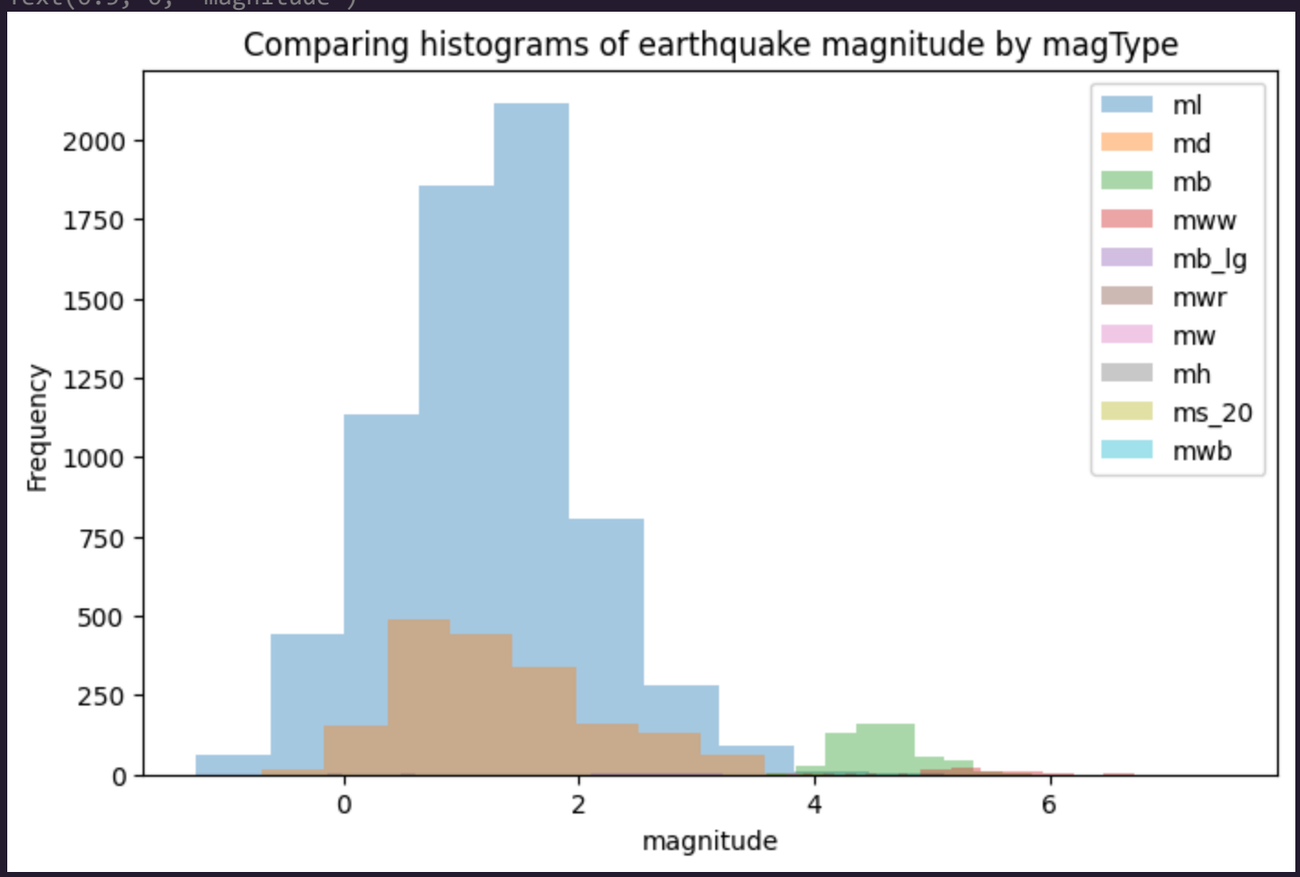
```
fb.volume.plot(
 kind='hist',
 title='Histogram of Daily Volume Traded in Facebook Stock'
)
plt.xlabel('Volume traded') # label the x-axis (discussed in chapter 6)
```

```
Text(0.5, 0, 'Volume traded')
```



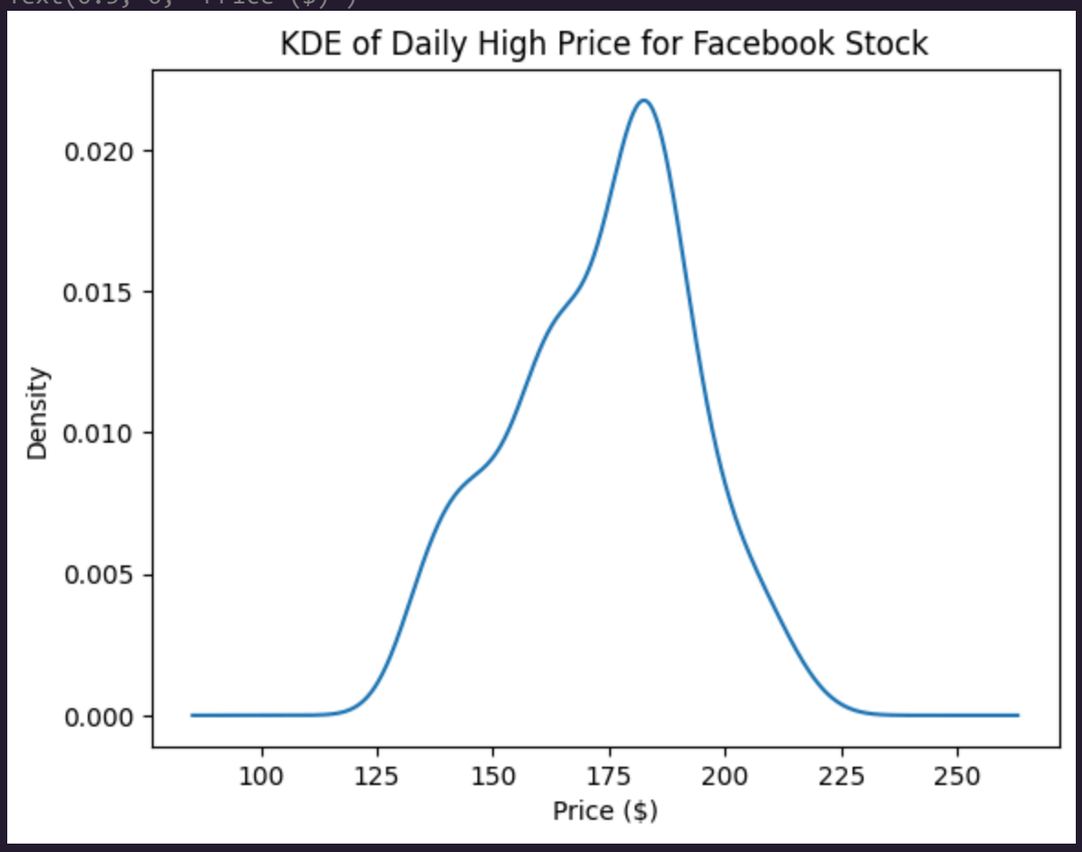

```
fig, axes = plt.subplots(figsize=(8, 5))
for magtype in quakes.magType.unique():
 data = quakes.query(f'magType == "{magtype}"').mag
 if not data.empty:
  data.plot(
    kind='hist', ax=axes, alpha=0.4,
    label=magtype, legend=True,
    title='Comparing histograms of earthquake magnitude by magType'
 )
plt.xlabel('magnitude') # label the x-axis (discussed in chapter 6)
```

```
Text(0.5, 0, 'magnitude')
```



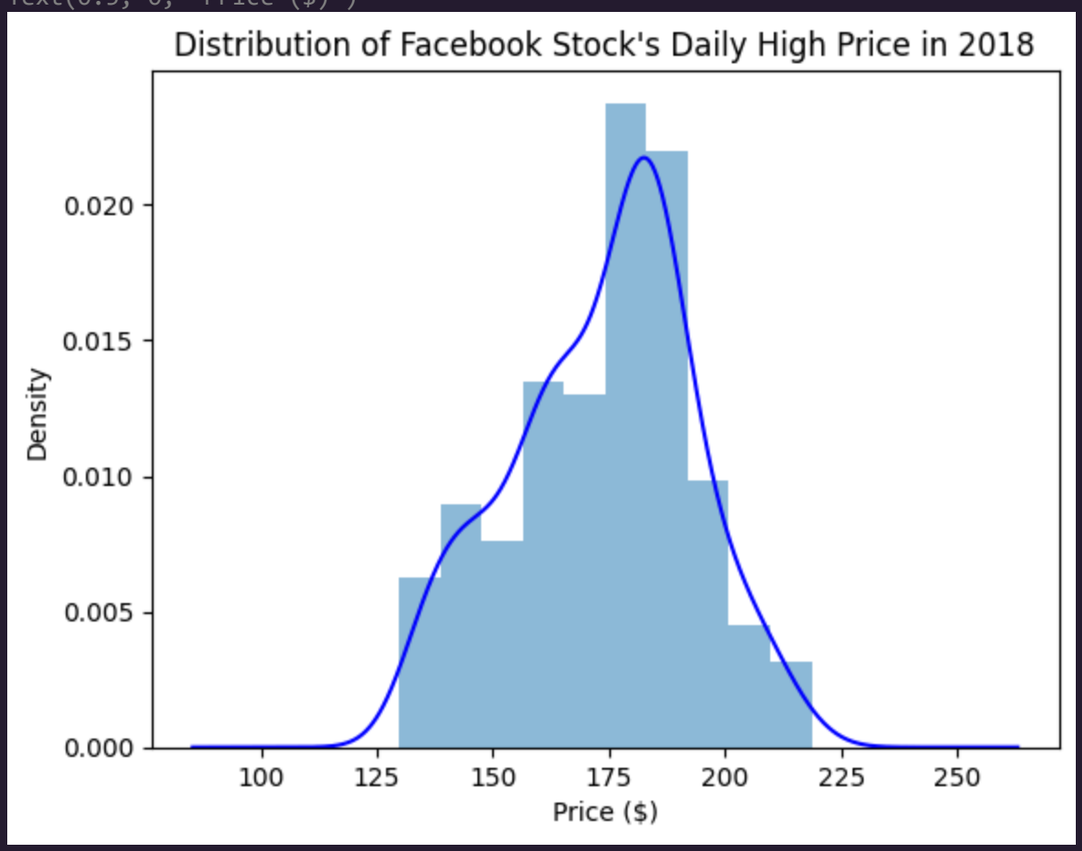
```
fb.high.plot(
 kind='kde',
 title='KDE of Daily High Price for Facebook Stock'
)
plt.xlabel('Price ($)') # label the x-axis (discussed in chapter 6)
```

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



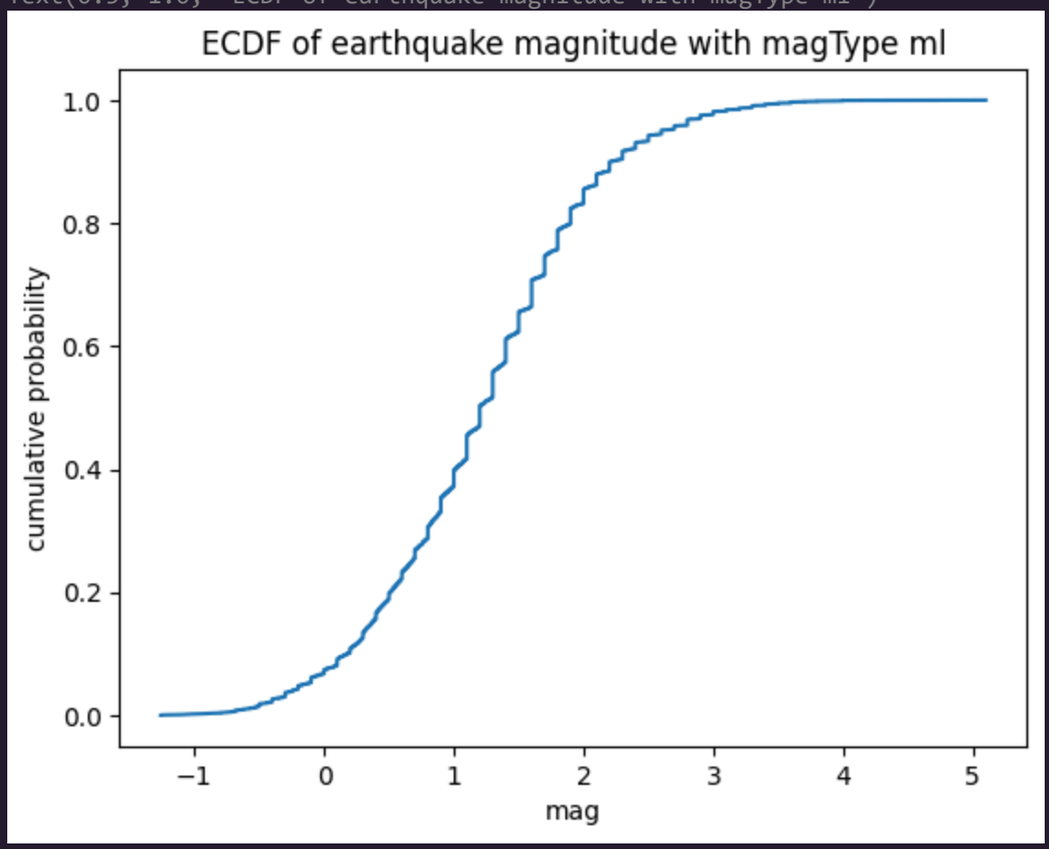

```
ax = fb.high.plot(kind='hist', density=True, alpha=0.5)
fb.high.plot(
 ax=ax, kind='kde', color='blue',
 title='Distribution of Facebook Stock\'s Daily High Price in 2018'
)
plt.xlabel('Price ($)') # label the x-axis (discussed in chapter 6)
```

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



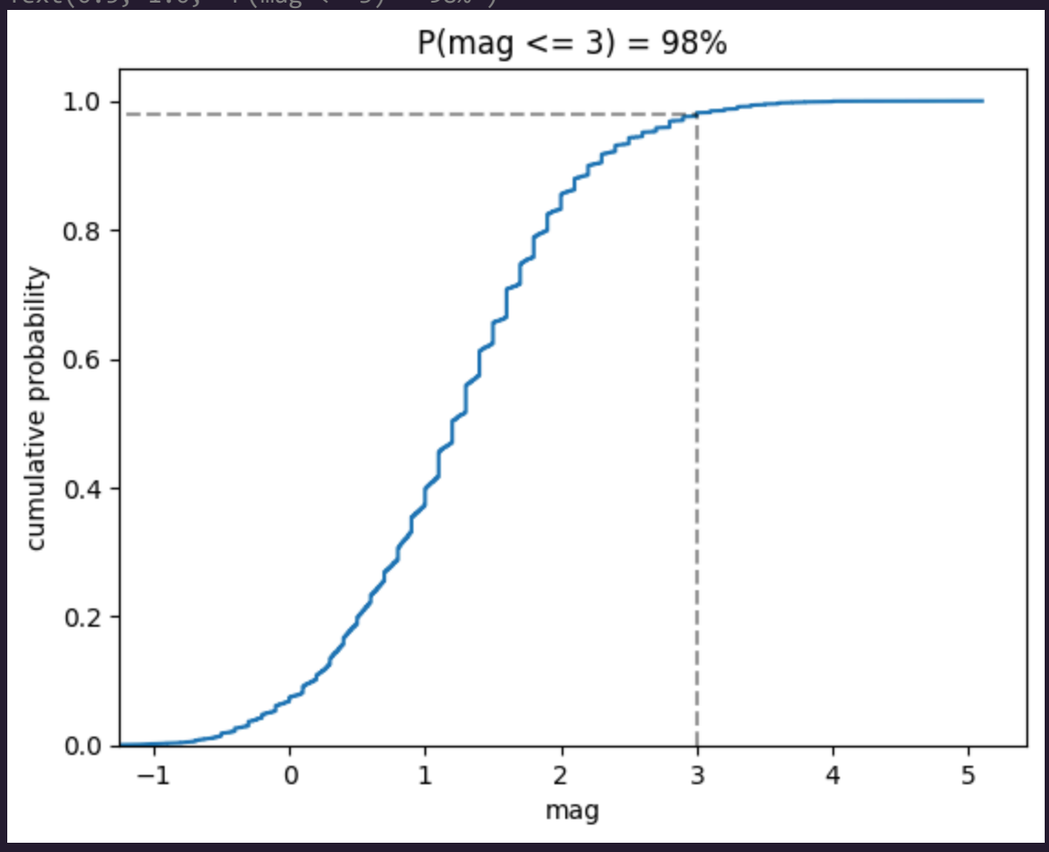
```
from statsmodels.distributions.empirical_distribution import ECDF
ecdf = ECDF(quakes.query('magType == "ml"').mag)
plt.plot(ecdf.x, ecdf.y)
# axis labels (we will cover this in chapter 6)
plt.xlabel('mag') # add x-axis label
plt.ylabel('cumulative probability') # add y-axis label
# add title (we will cover this in chapter 6)
plt.title('ECDF of earthquake magnitude with magType ml')
```

```
Text(0.5, 1.0, 'ECDF of earthquake magnitude with magType ml')
```



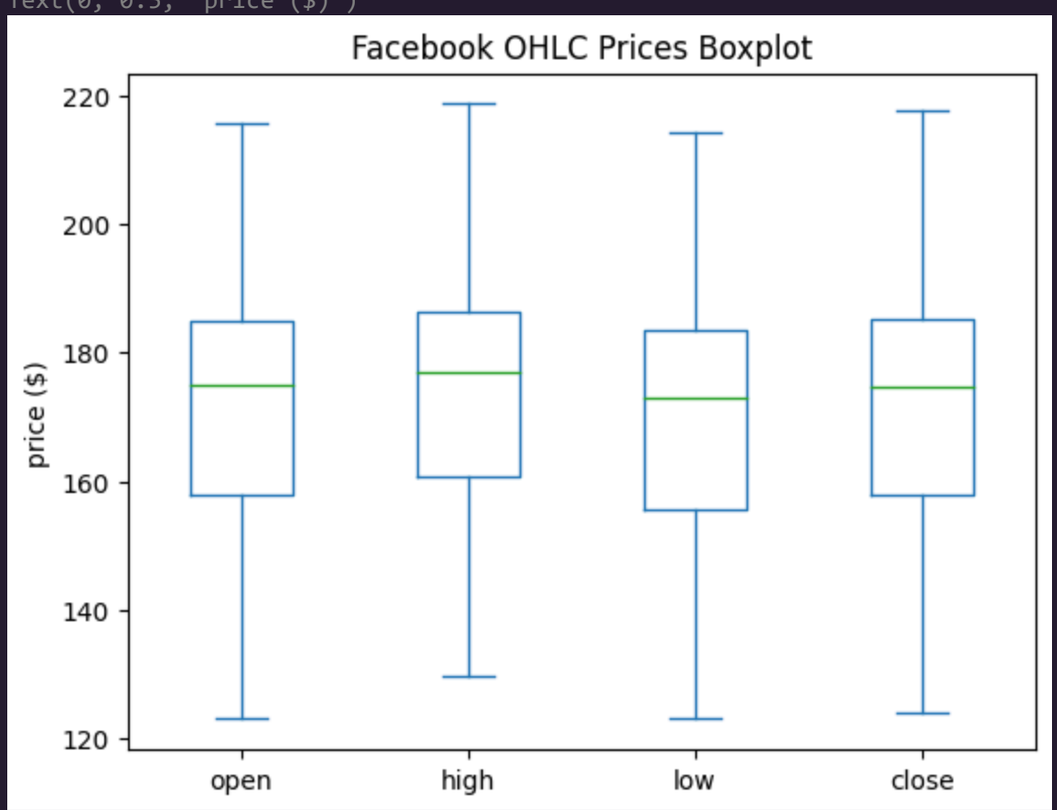

```
from statsmodels.distributions.empirical_distribution import ECDF
ecdf = ECDF(quakes.query('magType == "ml"').mag)
plt.plot(ecdf.x, ecdf.y)
# formatting below will all be covered in chapter 6
# axis labels
plt.xlabel('mag') # add x-axis label
plt.ylabel('cumulative probability') # add y-axis label
# add reference lines for interpreting the ECDF for mag <= 3
plt.plot(
 [3, 3], [0, .98], 'k--',
 [-1.5, 3], [0.98, 0.98], 'k--', alpha=0.4
)
# set axis ranges
plt.ylim(0, None)
plt.xlim(-1.25, None)
# add a title
plt.title('P(mag <= 3) = 98%')

```

```
Text(0.5, 1.0, 'P(mag <= 3) = 98%')
```



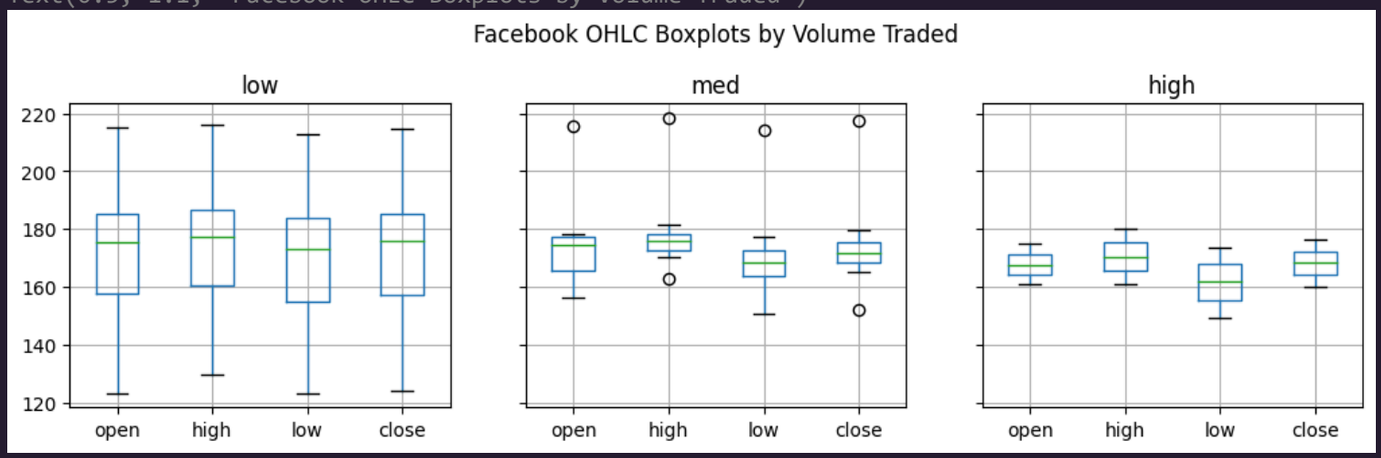

```
1 fb.iloc[:,:4].plot(kind='box', title='Facebook OHLC Prices Boxplot')
2 plt.ylabel('price ($)') # label the x-axis (discussed in chapter 6)
```

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



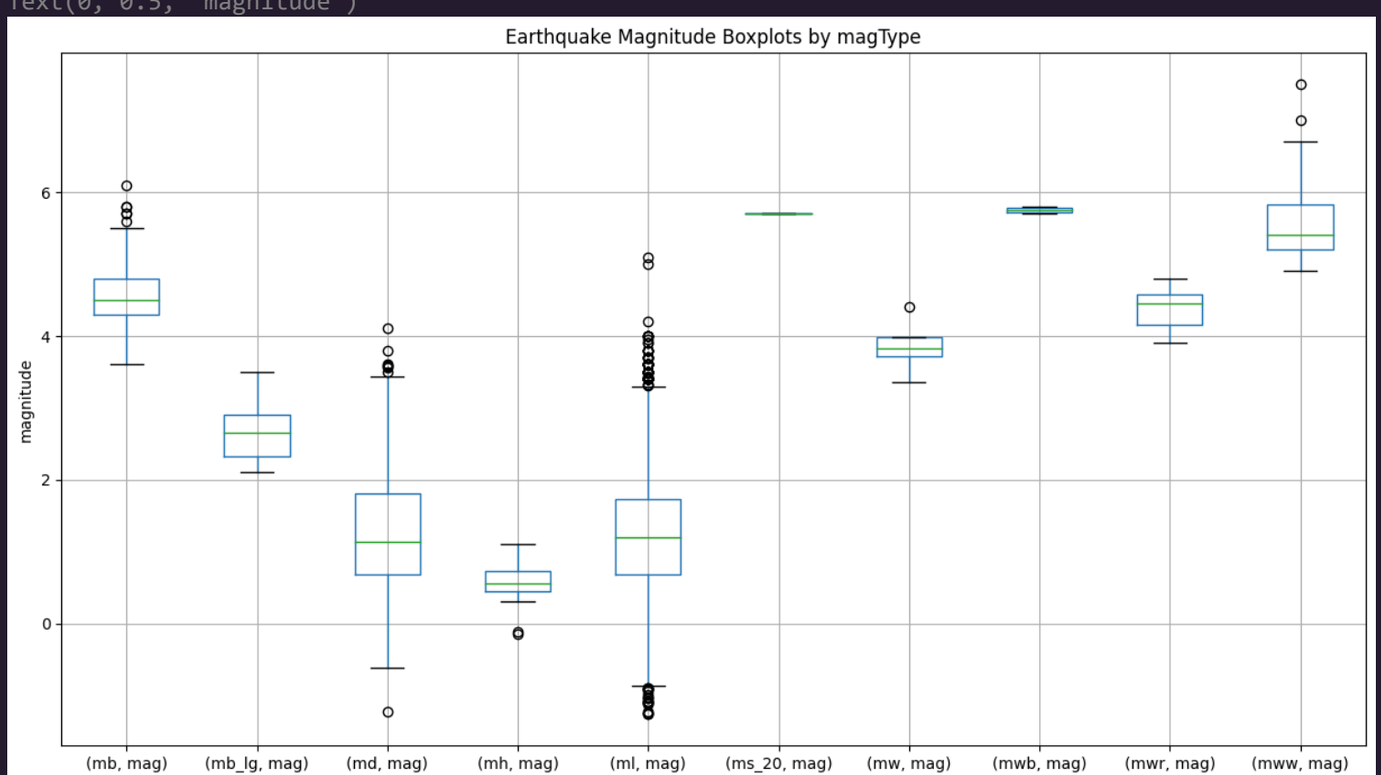

```
1 fb.assign(
2  volume_bin=pd.cut(fb.volume, 3, labels=['low', 'med', 'high'])
3 ).groupby('volume_bin').boxplot(
4  column=['open', 'high', 'low', 'close'],
5  layout=(1, 3), figsize=(12, 3)
6 )
7 plt.suptitle('Facebook OHLC Boxplots by Volume Traded', y=1.1)
```

```
Text(0.5, 1.1, 'Facebook OHLC Boxplots by Volume Traded')
```



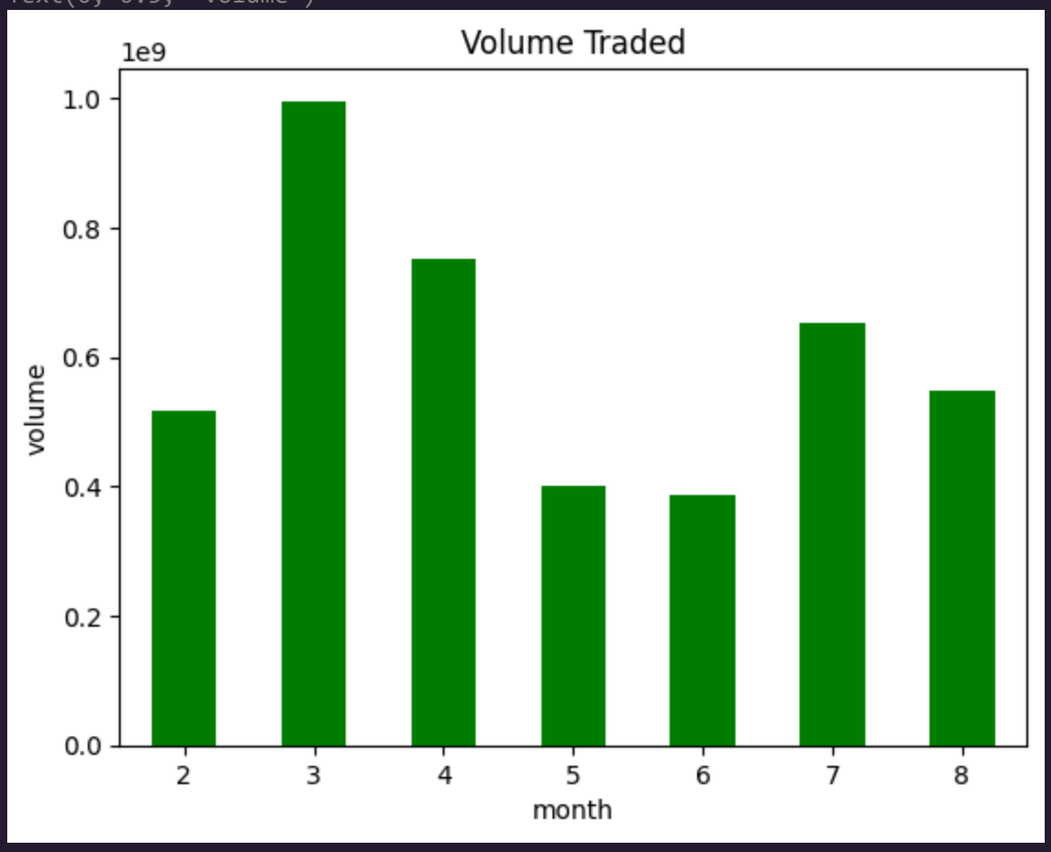

```
1 quakes[['mag', 'magType']].groupby('magType').boxplot(
2  figsize=(15, 8), subplots=False
3 )
4 plt.title('Earthquake Magnitude Boxplots by magType')
5 plt.ylabel('magnitude') # label the y-axis (discussed in chapter 6)
```

```
Text(0, 0.5, 'magnitude')
```



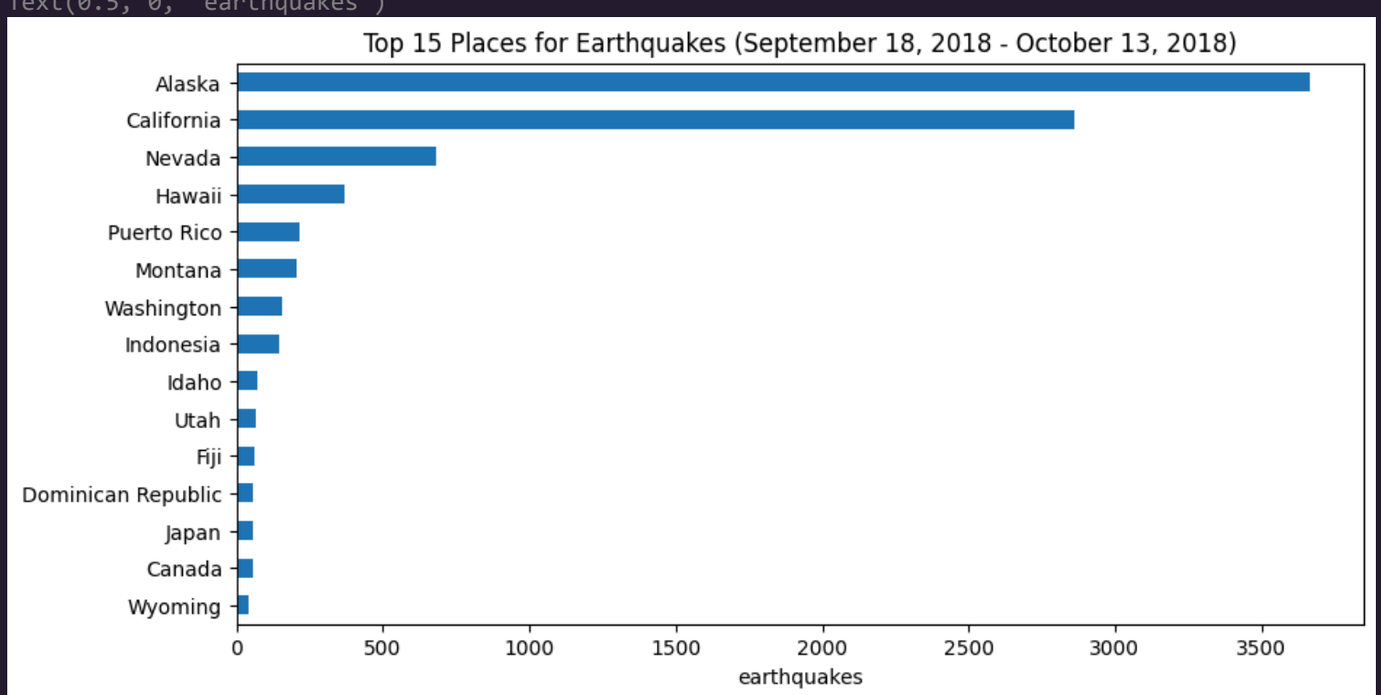

```
1 fb['2018-02':'2018-08'].assign(
2  month=lambda x: x.index.month
3 ).groupby('month').sum().volume.plot.bar(
4  color='green', rot=0, title='Volume Traded'
5 )
6 plt.ylabel('volume') # label the y-axis (discussed in chapter 6)
```

```
Text(0, 0.5, 'volume')
```



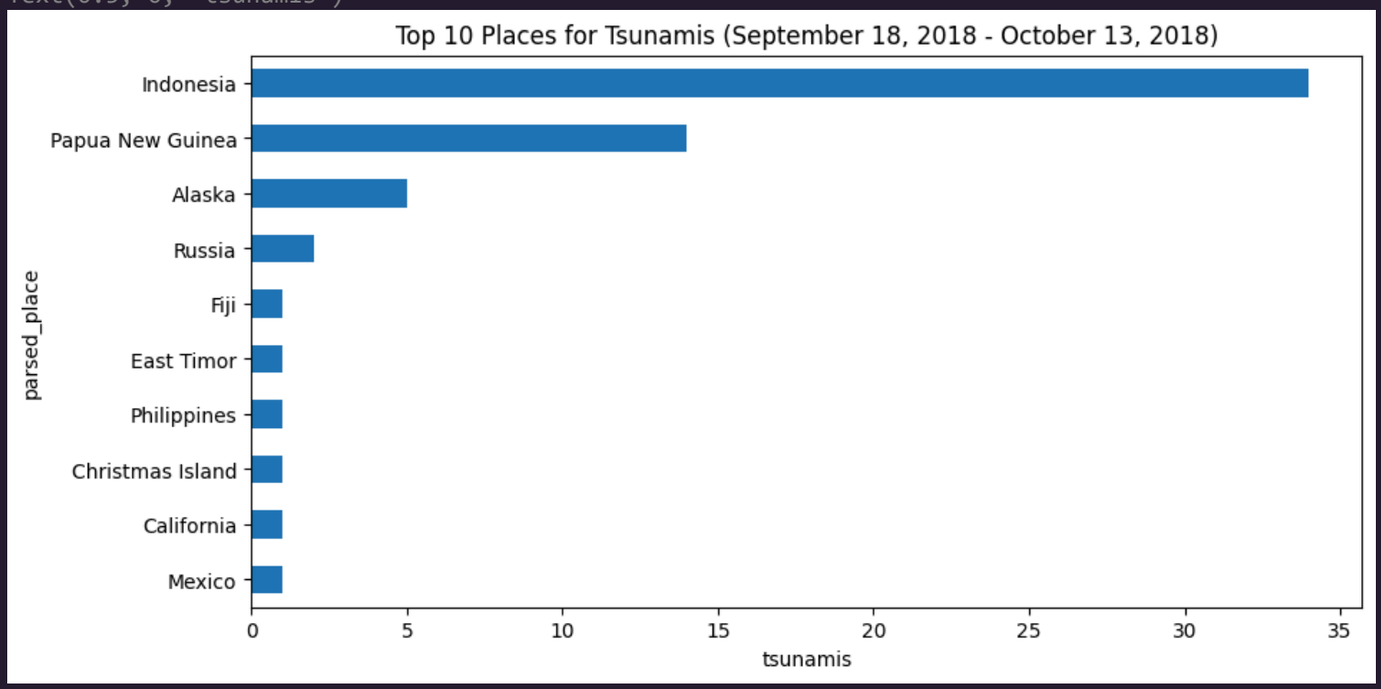

```
quakes.parsed_place.value_counts().iloc[14::-1,].plot(
 kind='barh', figsize=(10, 5),
 title='Top 15 Places for Earthquakes '\
 '(September 18, 2018 - October 13, 2018)'
)
plt.xlabel('earthquakes') # label the x-axis (discussed in chapter 6)
```

```
Text(0.5, 0, 'earthquakes')
```



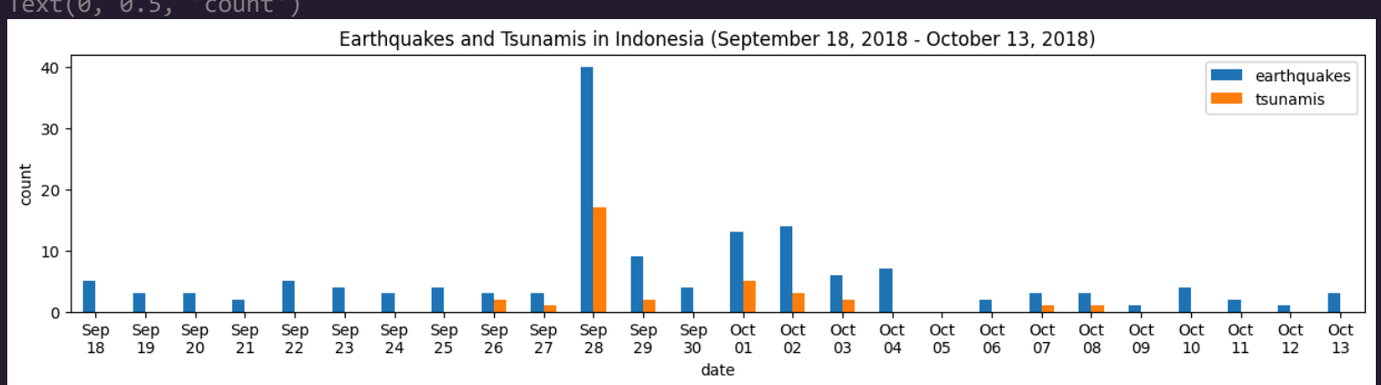

```
quakes.groupby('parsed_place').tsunami.sum().sort_values().iloc[-10::,].plot(
 kind='barh', figsize=(10, 5),
 title='Top 10 Places for Tsunamis '\
 '(September 18, 2018 - October 13, 2018)'
)
plt.xlabel('tsunamis') # label the x-axis (discussed in chapter 6)
```

```
Text(0.5, 0, 'tsunamis')
```




```
indonesia_quakes = quakes.query('parsed_place == "Indonesia"').assign(
 time=lambda x: pd.to_datetime(x.time, unit='ms'),
 earthquake=1
).set_index('time').resample('1D').sum()
indonesia_quakes.index = indonesia_quakes.index.strftime('%b\n%d')
indonesia_quakes.plot(
 y=['earthquake', 'tsunami'], kind='bar', figsize=(15, 3), rot=0,
 label=['earthquakes', 'tsunamis'],
 title='Earthquakes and Tsunamis in Indonesia '\
 '(September 18, 2018 - October 13, 2018)'
)
# label the axes (discussed in chapter 6)
plt.xlabel('date')
plt.ylabel('count')

```

```
<ipython-input-31-3671e7677b7a>:4: FutureWarning: The default value of numeric_only in DataFrameG
  ).set_index('time').resample('1D').sum()
Text(0, 0.5, 'count')
```




```
quakes.magType.value_counts().plot(
 kind='bar', title='Earthquakes Recorded per magType', rot=0
)
# label the axes (discussed in chapter 6)
plt.xlabel('magType')
plt.ylabel('earthquakes')
```

```
Text(0, 0.5, 'earthquakes')
```