What is financial trading

FINANCIAL TRADING IN PYTHON

Chelsea Yang
Data Science Instructor

datacamp

The concept of financial trading

Financial trading is the buying and selling of financial assets

Various financial instruments to trade:

- Equities
- Bonds
- Forex
- Commodities
- Cryptocurrencies

Why people trade

To make a profit by taking calculated risks

- Long positions: profit from upward price movement
- Short positions: profit from downward price movement

Market participants:

- Institutional traders
- Retail traders

Trading vs. investing

Trading:

- Shorter holding period
- Focus on short-term trends or price fluctuations
- Take both long and short positions

Investing:

- Longer holding period
- Focus on market fundamentals
- Take mostly long positions

Financial time series data

Time series data: a sequence of data points indexed in time order

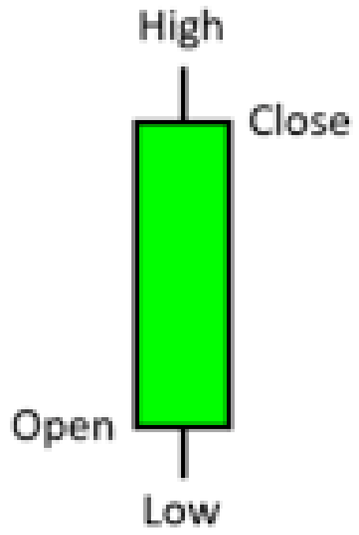
```
import pandas as pd
# Load the data
bitcoin_data = pd.read_csv("bitcoin_data.csv", index_col='Date', parse_dates=True)
print(bitcoin_data.head())
```

```
                Open      High       Low     Close       Volume
Date
2019-01-01  3746.71  3850.91  3707.23  3843.52  4324200990
2019-01-02  3849.22  3947.98  3817.41  3943.41  5244856835
2019-01-03  3931.05  3935.69  3826.22  3836.74  4530215218
2019-01-04  3832.04  3865.93  3783.85  3857.72  4847965467
2019-01-05  3851.97  3904.90  3836.90  3845.19  5137609823
```

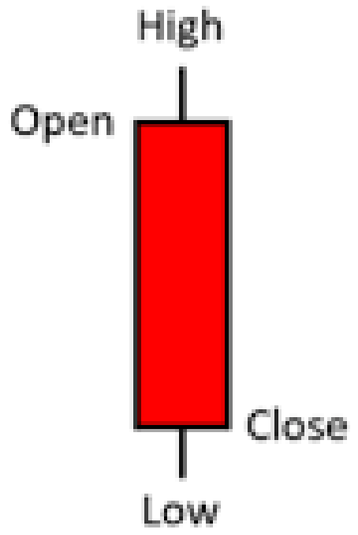
Plot line chart of time series data

```
import matplotlib.pyplot as plt
plt.plot(bitcoin_data['Close'], color='red')
plt.title("Daily close price")
plt.show()
```

Candlestick chart



Bullish candlestick



Bearish candlestick

- Each candlestick displays high, low, open, and close
- The color indicates bullish (rising prices) or bearish (falling prices) movement

Plot candlestick chart with Python

```
import plotly.graph_objects as go
# Define the candlestick
candlestick = go.Candlestick(
    x = bitcoin_data.index,
    open = bitcoin_data['Open'],
    high = bitcoin_data['High'],
    low = bitcoin_data['Low'],
    close = bitcoin_data['Close'])
# Create a plot
fig = go.Figure(data=[candlestick])
# Show the plot
fig.show()
```

Let's practice!

FINANCIAL TRADING IN PYTHON

Getting familiar with your trading data

FINANCIAL TRADING IN PYTHON

Chelsea Yang
Data Science Instructor

datacamp

Different types of traders

- Day Trader: holds positions throughout the day but usually not overnight
- Swing Trader: holds positions from a few days to several weeks
- Position Trader: holds positions from a few months to several years

Resample the data

```
Date                 Close
2019-11-29 04:00:00  1.1010
2019-11-29 08:00:00  1.1005
2019-11-29 12:00:00  1.0993
2019-11-29 16:00:00  1.1016
2019-11-29 20:00:00  1.1020
```

```
# Resample from hourly to daily
eurusd_daily = eurusd_h.resample('D').mean()
```

```
Date        Close
2019-11-25  1.10165
2019-11-26  1.10165
2019-11-27  1.10058
2019-11-28  1.10083
2019-11-29  1.10093
```

```
# Resample from hourly to weekly
eurusd_weekly = eurusd_h.resample('W').mean()
```

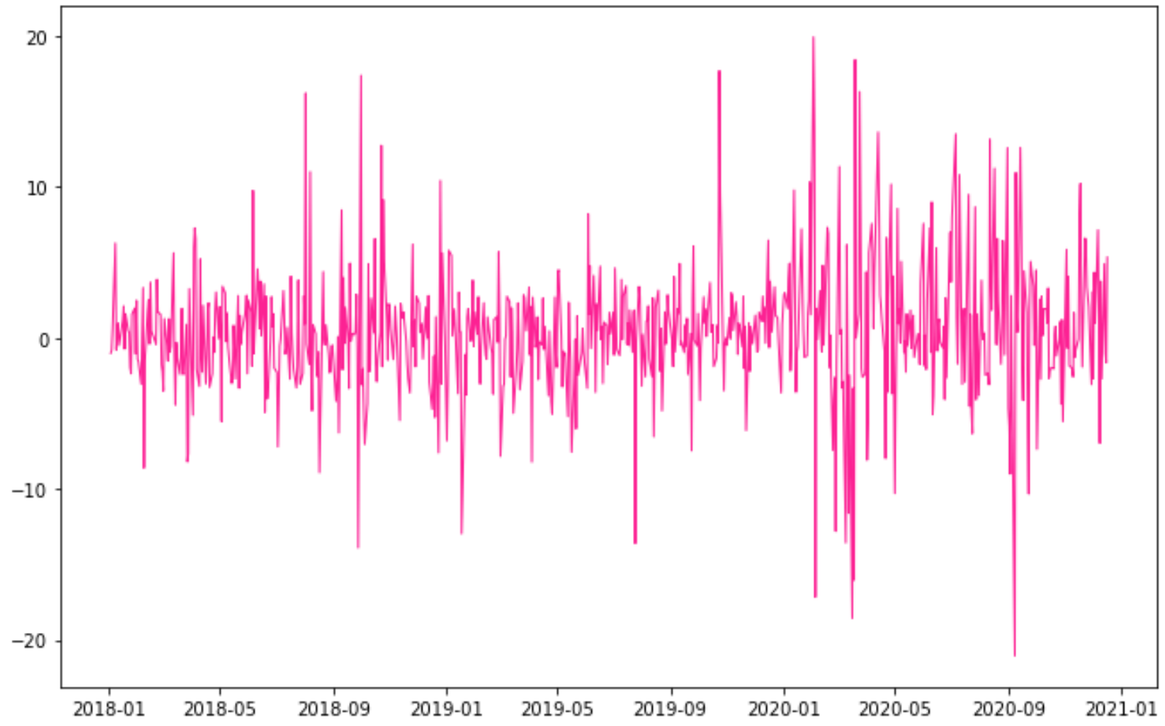
```
Date        Close
2019-11-03  1.11248
2019-11-10  1.10860
2019-11-17  1.10208
2019-11-24  1.10659
2019-12-01  1.10113
```

Calculate daily returns

```
# Calculate daily returns
stock_data['daily_return']
= stock_data['Close'].pct_change() * 100
```

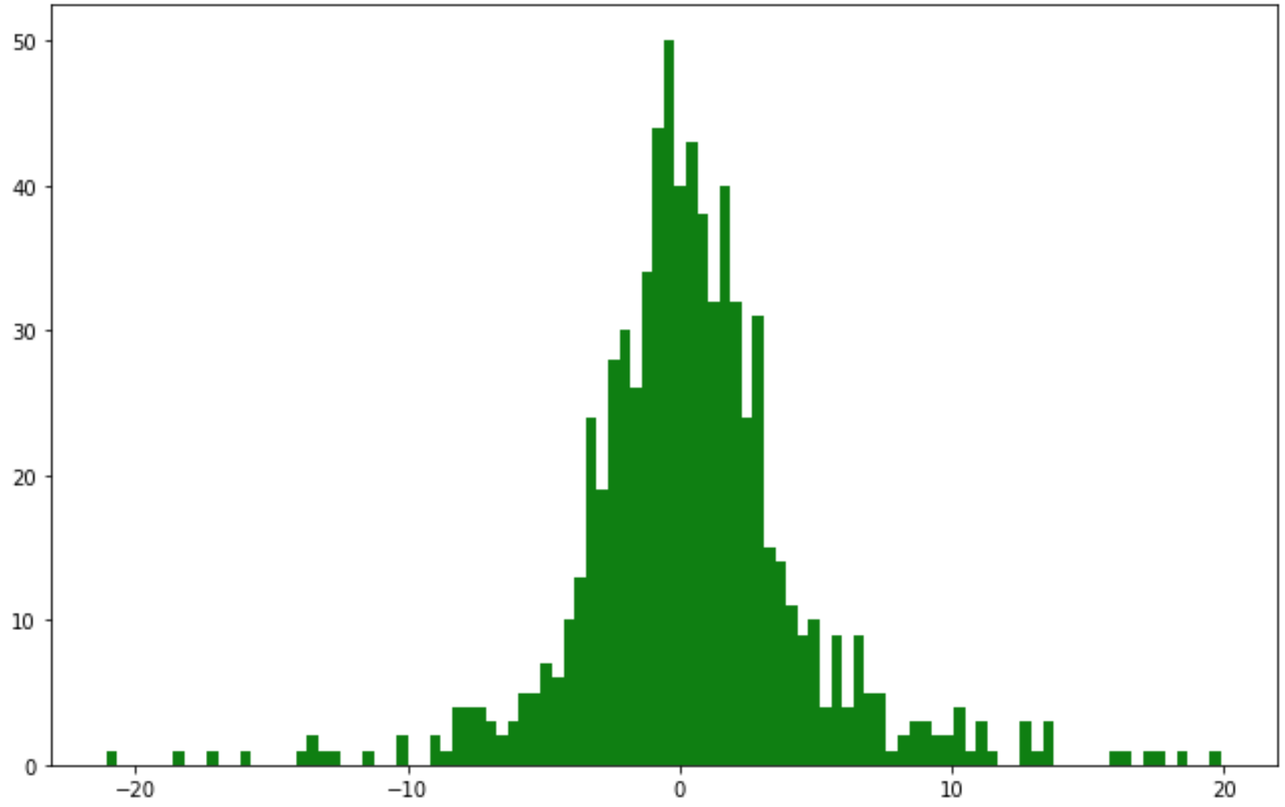
```
# Plot the data
plt.plot(stock_data['daily_return'])
plt.show()
```

Date	Close	daily_return
2020-12-11	609.99	-2.723779
2020-12-14	639.83	4.891883
2020-12-15	633.25	-1.028398
2020-12-16	622.77	-1.654955
2020-12-17	655.90	5.319781



Plot a histogram of daily returns

```
stock_data['daily_return'].hist(bins=100)
plt.show()
```



Data transformation

Technical indicators: various types of data transformations

Simple moving average (SMA): arithmetic mean price over a specified n-period

```
stock_data['sma_50'] = stock_data['Close'].rolling(window=50).mean()
```

```
             Close    sma_50
Date
2020-12-11  122.41   117.7474
2020-12-14  121.78   117.9226
2020-12-15  127.88   118.1502
2020-12-16  127.81   118.4432
2020-12-17  128.70   118.7156
```

Plot the rolling average

```
import matplotlib.pyplot as plt

plt.plot(stock_data['Close'],
         label='Close')
plt.plot(stock_data['sma_50'],
         label='SMA_50')
plt.legend()
plt.show()
```

Let's practice!

FINANCIAL TRADING IN PYTHON

Financial trading with bt

FINANCIAL TRADING IN PYTHON

Chelsea Yang
Data Science Instructor

datacamp

The bt package

A flexible framework for defining and backtesting trading strategies

- Strategy: a method of buying and selling financial assets based on predefined rules
- Strategy backtesting: a way to assess the effectiveness of a strategy by testing it on historical data

```
import bt
```

The bt process

- Step 1: Get the historical price data
- Step 2: Define the strategy
- Step 3: Backtest the strategy with the data
- Step 4: Evaluate the result

Get the data

```
# Download historical prices
bt_data = bt.get('goog, amzn, tsla',
                 start='2020-6-1', end='2020-12-1')
print(bt_data.head())
```

```
                goog         amzn        tsla
Date
2020-06-01  1431.819946  2471.040039  179.619995
2020-06-02  1439.219971  2472.409912  176.311996
2020-06-03  1436.380005  2478.399902  176.591995
2020-06-04  1412.180054  2460.600098  172.876007
2020-06-05  1438.390015  2483.000000  177.132004
```

Define the strategy

```
# Define the strategy
bt_strategy = bt.Strategy('Trade_Weekly',
                          [bt.algos.RunWeekly(), # Run weekly
                           bt.algos.SelectAll(), # Use all data
                           bt.algos.WeighEqually(), # Maintain equal weights
                           bt.algos.Rebalance()]) # Rebalance
```

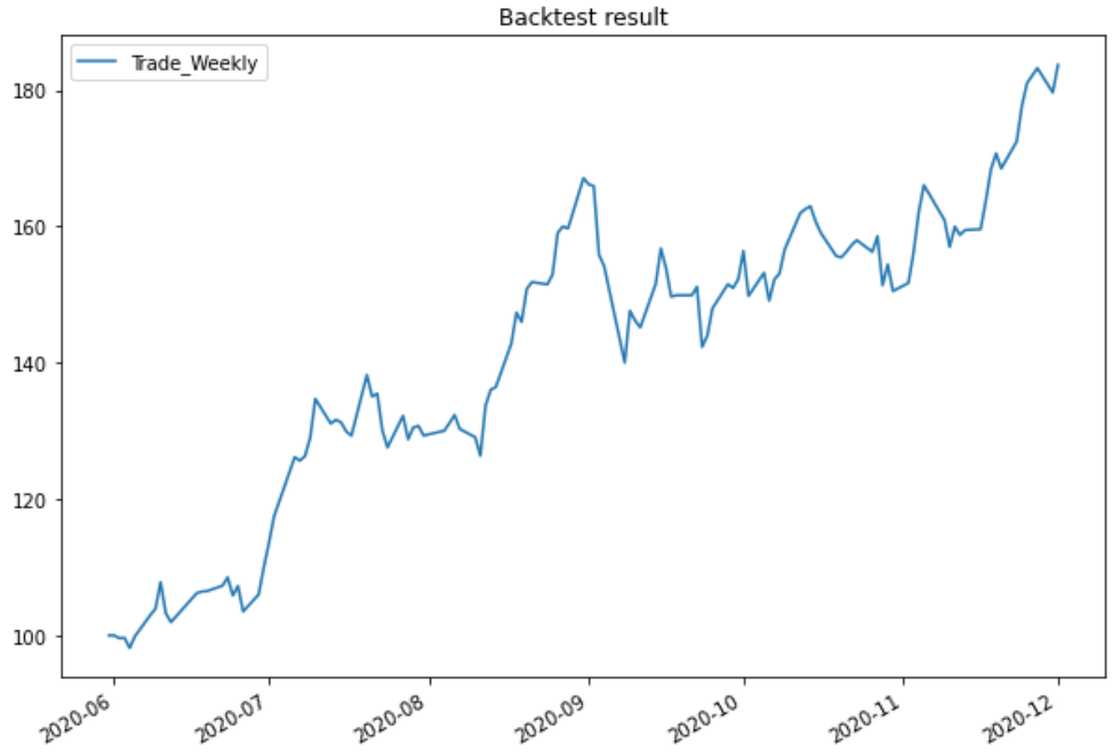
Backtest

```
# Create a backtest
bt_test = bt.Backtest(bt_strategy, bt_data)

# Run the backtest
bt_res = bt.run(bt_test)
```

Evaluate the result

```
# Plot the result
bt_res.plot(title="Backtest result")
```




```
# Get trade details
bt_res.get_transactions()
```

		price	quantity
Date	**Security**		
2020-06-01	**amaz**	0.015000	22222222.0
	goog	1431.819946	232.0
	tsla	179.619995	1855.0
2020-06-08	**amaz**	0.011200	5700757.0
	goog	1446.609985	-16.0
...	**...**	...	...
2020-11-23	**goog**	1734.859985	23.0
	tsla	521.849976	-132.0
2020-11-30	**amaz**	0.004900	458015.0
	goog	1760.739990	7.0
	tsla	567.599976	-27.0

Let's practice!

FINANCIAL TRADING IN PYTHON