Chapter 1

Foundations of Capital Markets

This chapter introduces the core concepts of modern financial markets and how they're represented in QuantConnect. We'll cover the modern US markets, data feeds, and the asset classes used in later chapters. Readers who are familiar with QuantConnect may skip this chapter.

Market Mechanics

The United States has 11 major stock exchanges. The two largest are the New York Stock Exchange (NYSE) and the National Association of Securities Dealers Automated Quotations System (NASDAQ). Trades on these exchanges are compiled by the Securities Information Processor (SIP) into a single data feed. This feed helps the Securities and Exchange Commission (SEC) determine the national best bid or offer (NBBO), which shows the best prices posted on public markets in the United States. When a new quote for more than 100 shares offers a better price, it is flagged as the NBBO. Quotes or trades involving fewer than 100 shares, known as odd lots, are excluded from this pricing. Figure 1.1 illustrates this flow.

Brokerages often send orders to market makers to be executed "off the market." Market makers executing these orders are required to provide fills within the NBBO price range. Furthermore, these off-market trades are reported to the Trade Reporting Facility (TRF) and eventually are included in the SIP data feed. Some brokers offer

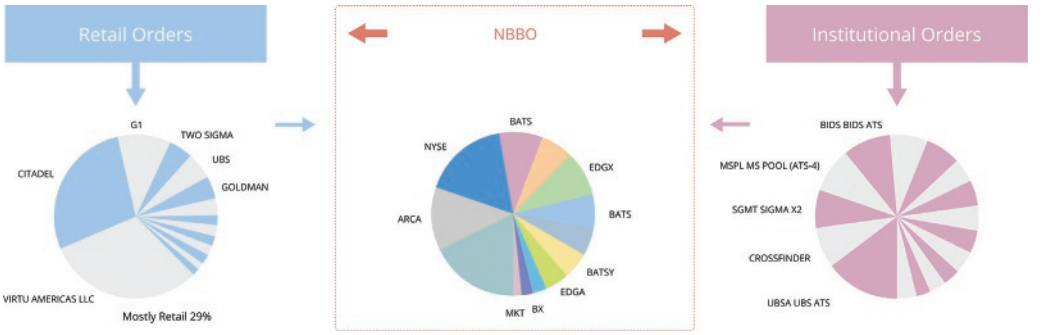

Figure 1.1 Flow of retail and institutional traffic across public and private markets, and the origin of national best pricing.

Direct Market Access (DMA), which allows your orders to be routed directly to a specific exchange. However, using DMA might not always get you the best national price for the asset, so it's important to be careful when using this option.

Market Participants

Trading Is the "Play"

If markets are theaters, then trading is the "play." Like a Shakespearean play, trading, especially algorithmic trading, is a highly coordinated and scripted activity. Comparing trading to the Bard's plays will probably make Shakespeare turn in his grave. But he will forgive this literary enthusiast for the forced metaphor.

The Stage and Basic Rules of Trading—The Limit Order Book

To stretch this analogy further, the stage of trading is the "limit order book," which is a ledger of some sort that lists limit order prices of a security in columns. On the left side is the column of bid prices, that is, prices traders are willing to buy a security at, and the amounts or "sizes" of the orders (for stocks, the sizes are typically in multiples of round lots of 100 shares; and for futures and options, in numbers of contracts). On the opposite side of this ledger is the column of ask prices, that is, prices traders are willing to sell a security at, and sizes. The prices are usually sorted from high to low from top to bottom, with the best bid and ask meeting in the middle. Bid prices cannot be higher than ask prices, that is, bid and ask prices do not cross, otherwise the buyers and sellers will be able to fulfill each other's order in a way that benefits one or both sides. Any buy orders that enter the market higher than current best ask price is effectively a market order and will be matched up to the size of the best ask price, and then the next best ask price and so on, until entire buy orders are filled in a process called "walk-the-book," or when remaining ask prices are above the best bid price. Because of this, a trader who posts limit orders will usually post bid prices below the best ask price and ask prices above the best bid price.

Because there is a spread between the best bid and ask prices, limit orders do not get filled right away, and indeed sometimes, not filled at all. To ensure immediacy of trades, in a sufficiently liquid market, a trader can post a "market order," which gets matched with the best bid for a market sell order and matched with best ask for a market buy order. If there is not enough size at the current bid and asks, the market sell and buy orders will walk the book as described previously: the market buy order will pay progressively higher prices; and conversely, the market sell order will accept progressively lower prices. If the size of the market order is large compared to sizes of available limit orders, the order will walk deeper (i.e., higher for buyer or lower for seller) in the order book, causing an immediate rise or decline in the trade price of the securities. This is a form of adverse price impact of trading.

Actors—Liquidity Trader, Market Maker, and Informed Trader

Now that the stage is set, let's introduce the actors, or more appropriately, the characters or roles in the play. Just like actors, traders can play multiple roles, sometimes in the same play or even at the same time.

Liquidity Trader

A "liquidity trader," also called "fundamental trader" or derogatively "noise trader," is a trader whose primary goal is to get in or out of a position for purposes other than profiting from advantaged information (not always insider information). For example, a mutual fund manager decides to rebalance her stock portfolio to match a benchmark index. The trader acting on the instruction of the fund manager is a liquidity trader. The same fund manager may be a macro forecaster, sector specialist, or stock picker basing her trades on publicly available information and deciding to buy or sell some stocks to profit from her mosaic view. Still, such a fund manager does not possess any advantaged information. The trader who executes her trades may be respectfully called a "fundamental trader," even though fundamentally she is no different from a liquidity trader. Finally, you also have undisciplined traders who are trading for the sake of trading—and we can safely call them "noise traders". Whether the traders are fundamental or noise, their objective is to complete buy or sell orders with no advantaged information.

Market Maker

If a liquidity trader wants liquidity, who is there to pour him a drink? Well, it could be the liquidity trader on the other side of the trade. It is likely that most liquid stock transactions occur by matching simultaneous market orders on opposite sides of the trade. And much of those orders are matched by your broker dealer before they reach the stock exchange. This is done by internally "crossing" or "netting" the orders, usually done at the mid-price, that is, halfway between best bid and ask in the limit order book.

If there is not enough market order from the opposite side, a market order will "hit the bid" or "lift the offer." It will walk the book if there is not enough size at the best bid or ask, as we discussed previously. Here, it is the other limit orders that will fulfill the liquidity-seeking market order. Exchanges will pay a rebate on a filled limit order to reward the trader for liquidity.

Strategies that facilitate trading and improve transaction prices, or immediacy, are strategies typically deployed by a "market maker." Traditionally, when trading was conducted on the floor of a stock exchange (e.g., NYSE), dedicated market makers were physically located in booths and their jobs were to match trades for their assigned list of stocks. They stood ready when the listed buy and sell (limit) orders were not sufficient to clear the market of open orders (e.g., when the market was at a standstill because the best bid and ask prices were too far apart). In situations like that, it was the market maker's job and her opportunity to post orders that get in between the best bid and ask prices to encourage liquidity traders to transact at her better prices. In return, the market maker would profit from the bid and ask spread, that is, she bought at her bid price and sold higher at the current or improved ask price. The previous narrative is in the past tense because the majority of market-making activities have shifted to electronic trading platforms. Some floor trading still exists for less liquid stocks and other asset classes. For example, the Chicago Board of Options still operates a pit to trade equity and equity index options, while the Chicago Mercantile Exchange ceased pit trading in March 2020 due to COVID-19 and decided the closure would be permanent.

In the modern era of electronic trading, market makers are now almost exclusively algorithmic and high-frequency trading programs. There are still traders whose primary or exclusive job is to make markets in certain stocks, but anyone can make markets intentionally or unintentionally by deploying strategies that improve outcomes—better prices and immediacy—for liquidity traders.

Informed Trader

Wait, a market maker helps only liquidity traders? What about other types of traders? Doesn't the market maker help everyone? Yes, market-making activity helps everyone, except other market makers who are competing against each other. That being said, market makers are more wary of "informed traders" than other market makers. To skirt the controversy about insider trading, we shall broadly define informed traders as ones who possess advantaged or privileged information that is only known to a small number of agents and is not released to the public. As we discussed previously, a trader or fund manager might have her unique opinion about the future outcome of financial markets and of individual securities. And she might have superior forecasting or analytical skills. But if she does not have advantaged information that others don't have, she is not considered an informed trader. On the other hand, advantaged information does not always have to be about the fundamental values of financial securities. For example, say a broker has the knowledge that a particular fund manager is about to make a large block trade that will have a significant impact on the market price. A trader who has this information can profit, illegally, by trading ahead of the fund manager, in a scheme called front-running.

It is not this author's job to argue the benefit or harm of insider trading or trading on privileged information. The point is that an informed trader has a huge advantage over other traders, as she knows about future outcomes with much greater certainty than other traders do. Those uninformed traders include both liquidity traders and market makers. Even a market maker with high-frequency trading rigs cannot win against low-tech informed traders. It is very difficult to know whether the counterparty is an informed trader. This means that market makers need to adjust their strategies to incorporate the risk of trading against informed traders.

AI Actors Wanted!

One of the exciting promises of machine learning and artificial intelligence (AI) is that we can train an algorithm to detect patterns of activities by informed traders. It is no longer humanly possible to sift through the terabytes of trading data generated each day by markets, not to mention analyzing and recognizing tradable patterns. Even if a human can recognize patterns, he will not be fast enough to react to them. Gone were the days when traders could sharpen their stock picking skills and find trading opportunities by studying the stock tables in the *Wall Street Journal* or the tiny thumbnail sized price charts on *Investor's Business Daily*. Therefore, it has become critical to rely on fast computers running powerful machine-learning algorithms to process and analyze market data sufficiently fast in order to detect and act on tradable signals.

Data and Data Feeds

As market participants enter bids, asks, or execute trades, an event is created that is broadcast through the market. Over millions of traders and investors, this creates a constant stream of events during the trading day and an enormous volume of data. Interestingly, the stream tends to drop off around lunchtime and picks back up around 3 pm when traders tune in for market close. The market open and close volumes can be two to three times mid-day volume.

Until the 1960s, this stream of trades and quotes was written to reels of paper tapes for posterity, creating a ticker tape. The saying "replaying the tape" comes from repeating those streams as they appeared in order.

```
self.time            # Time in backtesting and live trading
```

If the data streams are recorded as they appear, despite any potential errors or imperfections, they become a point-in-time dataset. Point-in-time data is a raw representation of the data at the precise moment it was captured but may include errors in the feed connectivity or exchanges themselves.

A classic regular example of this is the late reporting of trades performed. Although exchanges require participants to report trades within 90 seconds, some may report much later than that, at prices from trades that occurred earlier in the day. This appears like a price discontinuity and can be quite disconcerting for new quants who don't take care to filter them out.

Recently, some rapid market declines were exacerbated by algorithmic trading programs. The most famous is the 2010 Flash Crash, which started at approximately 2:30 pm and lasted about 35 minutes on May 6, 2010 (see Figure 1.2). In the aftermath of the 2010 Flash Crash, stock exchanges announced new trading curbs or circuit breakers. Several exchanges led by the NYSE and NASDAQ began retroactively canceling trades that occurred at very low prices during crashes. These cancelations occur minutes to hours after the trades and are impossible to know beforehand.

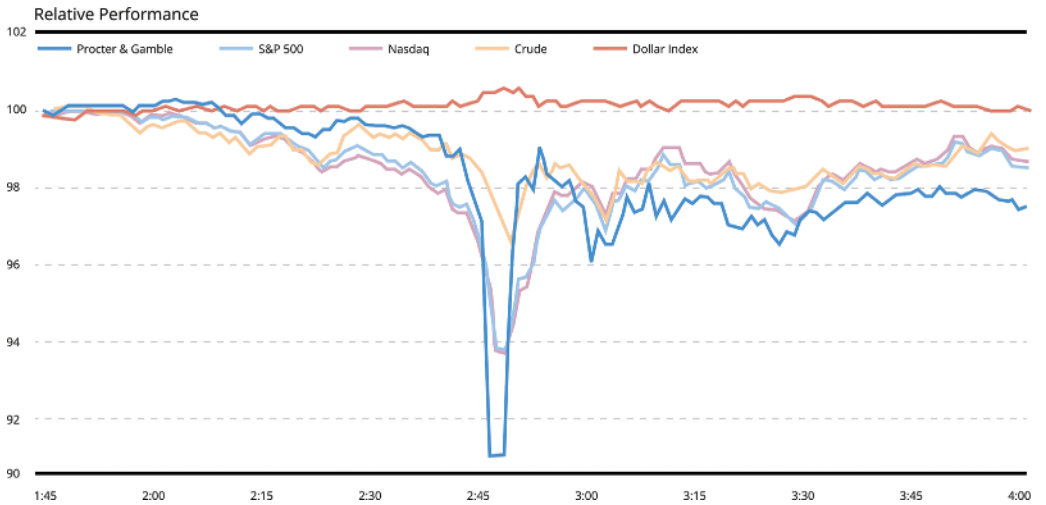

Figure 1.2 Relative performance of various assets during the market crash on May 6, 2010.

Market data feeds generate three primary data types of tick data:

Trade ticks—A sales report of an order filled, along with flags to convey information about the order, such as the exchange venue and delay.

Quote ticks—An offer to buy or sell a specific quantity of shares for a price. The difference between the best or highest bid prices (offers to buy) and the best or lowest asking price (offer to sell) forms the bid-ask spread. For US markets, the best quotes in the country form the National Best Bid or Offer (NBBO). These ticks are broadcast with a special flag to inform the market of the current best prices.

In QuantConnect, this data is delivered on each data event as a list of ticks. Note, "tick_type" is an attribute that identifies whether the data point is a trade tick or a quote tick.

```
def on_data(self, slice):
    ticks = slice.ticks.get(self._symbol, [])    # Empty if not found
    for tick in ticks:
        price = tick.price                       # Price limit or trade
        trade_or_quote = tick.tick_type          # TickType.TRADE or QUOTE
```

Consolidated Data—Tick data creates billions of events per day. It consumes a lot of disk space and is slow to process. To make the dataset smaller and easier to perform research on, ticks can be aggregated into bars. Most commonly, aggregations represent prices (open, high, low, close), volume traded, last bid, last ask, and so on, over a fixed interval of time. QuantConnect consolidates the market's stream of ticks into trade and quote bars (see qnt.co/book-consolidate). A trade bar represents the sale prices over a period, while quote bars represent the bids and asks aggregated over a bar period (Figure 1.3).

```
def on_data(self, slice):
    trade_bar = slice.bars.get(self._symbol)           # Fetch trade bar
    quote_bar = slice.quote_bars.get(self._symbol)     # Fetch quote bar
```

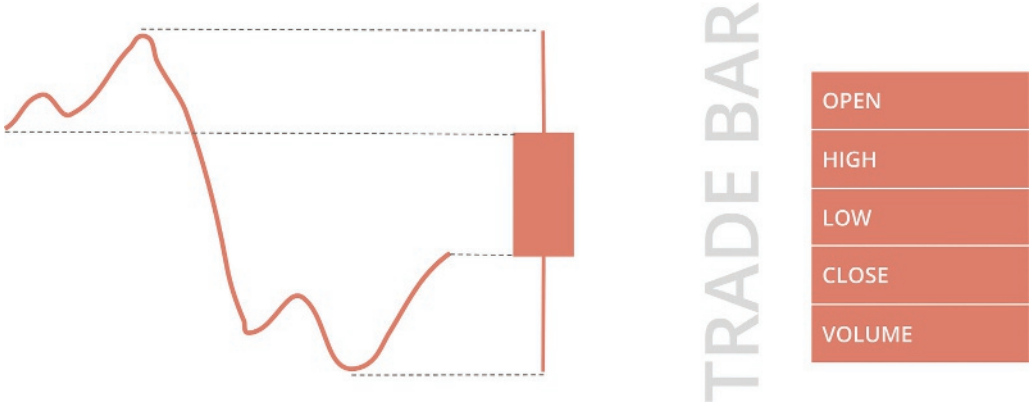

Figure 1.3 Trade bar formation and properties.

Each quote bar carries child bid and ask bars as properties, representing the aggregation of bid and ask data over a fixed interval. They are also in the form of an Open-High-Low-Close (OHLC) bar. In QuantConnect, when quote data is available for an asset, it is used for the modeling of order fills as it is a more accurate representation than using the previous trade price (Figure 1.4).

```
quote_bar.bid.close          # Close of aggregated bid price bar
quote_bar.ask.close          # Close of aggregated ask price bar
```

Custom and Alternative Data

In addition to price, a multitude of other data sources are available to provide insight into the movements of asset prices. Broadly, this category of data is called alternative data and includes imaging, real estate, weather, shipping, regulation, and a suite of customer tracking, including geolocation, reviews, sentiment, and transactions.

QuantConnect hosts more than 60 datasets (qnt.co/book-datasets) and frequently onboards new ones. Data can be added to the strategies with a couple of lines of code. The following example delivers news articles about Apple from a streaming news service, TiingoNews:

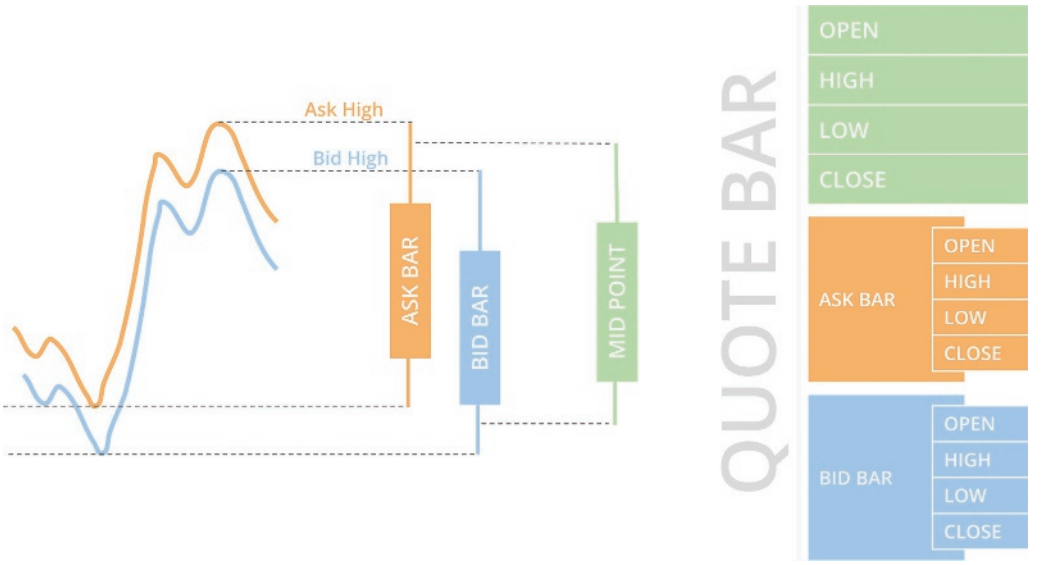

Figure 1.4 Quote bar formation and properties.

```
self._aapl = self.add_equity("AAPL", Resolution.MINUTE).symbol
self._dataset_symbol = self.add_data(TiingoNews, self._aapl).symbol
```

Generating tradable signals from alternative data is a science, and we'll go into more detail on how to analyze these datasets in later chapters. If QuantConnect does not have the data you need, you can upload custom datasets by defining the data format as a class and using the `add_data` method to add it to your strategy. In the following snippet, we're passing a custom class `MyFactorDataset`, which will define the properties of the data and parse a source CSV file:

```
self._custom_symbol = self.add_data(MyFactorDataset, "Factors").symbol
```

Brokerages and Transaction Costs

Brokers act as intermediaries between the capital markets and clients. They hold client assets, clear trades from the exchanges, and ensure clients remain within margin limits, enforcing SEC and FINRA margin rules. Brokerages typically clear and settle their trades with clearing firms. The majority of institutional and retail trades are executed through brokerage firms. We will not discuss the functions of the clearing firms in this book as they do not concern the readers here.

Brokers have varying fee structures, supported assets, and order types (qnt.co/book-order-types). Some brokerages route to market makers to fill trades, earning rebates from directing order flow to a specific market maker. Some may fill orders internally in a process called "netting" with other client orders. All internally filled orders must be executed within the national best bid-ask prices from public markets.

Brokerages can support many different types of orders, including some proprietary order types that seek optimal fills. Other brokerages have nuanced limitations, such as the inability to update an order once it has been placed.

QuantConnect implements 12 order types. Following are examples of two commonly used order types that are used in the later examples:

```
# On exchange open attempt to fill in the opening auction
self.market_on_open_order(symbol, quantity, tag, order_properties)

# On hitting a stop_price, place market order.
self.stop_market_order(symbol, quantity, stop_price, tag, order_properties)
```

Note, some order types may not be supported by the brokerage. In QuantConnect, all orders return an *order ticket* (qnt.co/book-order-ticket), which can be used to update or cancel an order before it is filled. Think of this as a coat ticket that lets you retrieve your coat from the cloakroom.

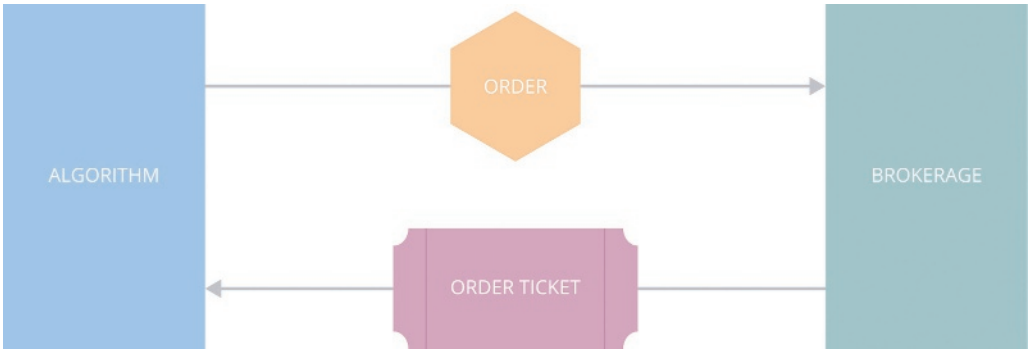

Figure 1.5 Order messaging between the algorithm and brokerage.

Transaction Costs

In trading, there are many types of transaction costs. Some are explicit, such as fees from the brokerage (qnt.co/book-fee-model) and taxes, and others are more implicit—incurred simply from the act of trading. For quantitative research, it's important to understand where these costs come from, so you can model them—and optimize your trading to reduce these costs.

Trading Fees

Brokers charge fees per order or share, which are charged directly to your brokerage account. In QuantConnect, this can be modeled with fee models to simulate specific trade fees or, more generally, a brokerage model that enforces a specific broker's limitations and costs.

```
# Setting a custom fee model of $1 for this security
security.set_fee_model(ConstantFeeModel(1))
```

```
# Setting all the fees and limitations of a broker
self.set_brokerage_model(BrokerageName.INTERACTIVE_BROKERS_BROKERAGE)
```

Bid-Ask Spread

For market orders, the most substantial of these implicit costs is the bid-ask spread: the difference between the best bid and ask prices (usually estimated from the NBBO, covered earlier) of an asset in the market. When a market order is filled, it crosses the spread and fills at the best available price (Figure 1.6). Some practitioners optimistically assume fills will occur at the last trade sale price, which is generally somewhere in the middle of the spread. In live trading, the spread can significantly impact profitability and should be accounted for in research.

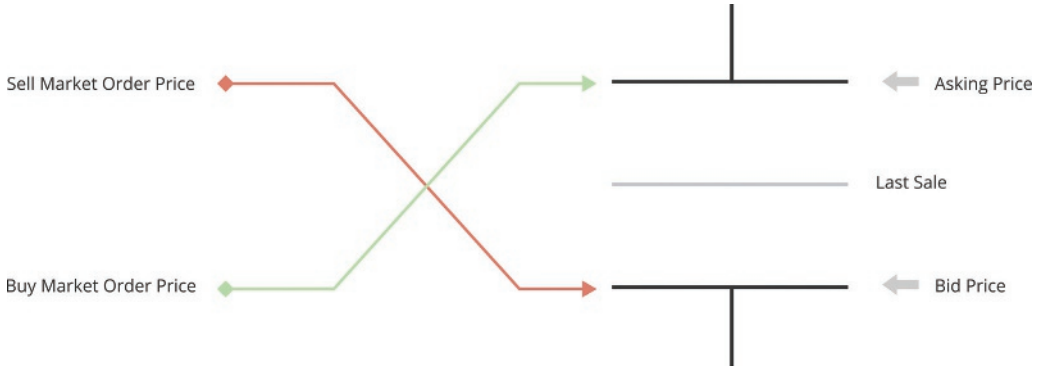

Figure 1.6 Spread crossing: sell market orders fill at the bid price, and buy market orders fill at the ask price. The last sale can be at the bid, at the ask, or somewhere in between them.

In QuantConnect, orders are filled using quote data, accounting for this spread. If the default fill behavior doesn't fit your use case, you can customize the behavior with plugins.

```
security.set_fill_model(ImmediateFillModel())
```

Slippage

Trade slippage (qnt.co/book-slippage-model) is the difference between the expected execution price and the final fill price for your trade. The most common cause for small orders, on liquid assets, is time as the asset prices change while the order is filled (Figure 1.7).

For illiquid assets or large orders, slippage can come from the temporary or permanent price impact of the trade order itself. A direct mechanism of price impact is through a process called "walk-the-book," which we discussed previously. The order book combines bid-ask prices at multiple levels with indicated sizes. A large market order to buy (i.e., an order to be executed immediately at the best ask price in the order book) will be filled with the best ask price first, then sequentially filled with higher and higher ask prices until the entire order is filled. This causes the traded price to move up by

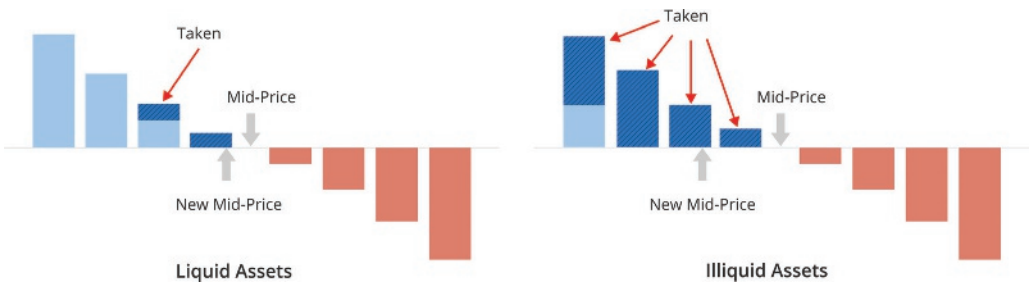

Figure 1.7 Slippage costs are different between liquid and illiquid assets.

"walking the book". The sell market order will walk the book downward in price following the same logic. The price impact can be temporary for liquid markets, as market makers replenish the order book at prices close to the original price before the market order arrived. However, if market makers suspect informed trader activity, they will replenish the order book with higher or lower prices near the last traded price (or worse). In this case, the price impact becomes permanent. A temporary price impact can be ameliorated by trading in smaller increments, while an anticipated permanent price impact will require more sophisticated trading algorithms to disguise the actual size of the trade.

QuantConnect assumes assets are liquid by default but offers several alternative models to customize more nuanced fill behavior:

```
# Assume instant fill at top of order book
security.set_slippage_model(NullSlippageModel())

# Model-estimated market impact in fills
security.set_slippage_model(MarketImpactSlippageModel(self))
```

Security Identifiers

Uniquely identifying assets is important to track them over time reliably. In equities, corporate events, such as renames, mergers, and exchange delistings, can render backtests misleading and inaccurate. Spotting these issues can be challenging if you're trading a portfolio of hundreds of stocks. There have been many unique asset identifiers developed in the last few decades to assist with asset tracking:

- CUSIP—A proprietary system limited to the United States and Canada that covers several asset classes. CUSIP is an eight-digit alphanumerical string, or nine digits, if it includes a "check-digit" encoded from the first eight digits.
- FIGI—Bloomberg's free, proprietary database look-up service.
- ISIN—For most US and Canadian assets with CUSIPs, the ISIN is a concatenation of the country code, for example, US, CA, and the eight-digit CUSIP.

A quant trader can access these identifiers by subscribing to a third-party asset price dataset that contains these identifiers such as Algoseek—or subscribe to identifier datasets directly from the providers. However, there are some key limitations to using these identifiers:

- They are proprietary datasets that often require expensive and restrictive licenses.
- Identifier datasets may not be self-contained and require a parallel database to look up information about the company.
- Coverage of different asset classes or country markets is limited.

QuantConnect solves these limitations by implementing an open-source encoding technique. The identifier (see qnt.co/book-security-id) is a hash of data required to "fingerprint" the asset, making it entirely self-contained, so no database lookup is required. This identifier is called the "Symbol," which supports up to 99 asset classes in 255 countries and handles derivative asset types. It has no licensing fee.

In 2014, Google performed a ticker swap and initial public offering (IPO) of Class C shares (qnt.co/book-goog-ipo) without voting rights. They listed GOOCV and GOOAV as temporary assets to facilitate the swap, and then flipped the tickers so the former Class A share ticker was used for class C (Figure 1.8).

```
self._goog = self.add_equity("GOOG").symbol     # Store the symbol for trading
self._googl = self.add_equity("GOOGL").symbol
self.debug(str(self._goog.id))                  # Prints "GOOCV VP83T1ZUHROL"
self.debug(str(self._googl.id))                 # Prints "GOOG T1AZ164W5VTX"
```

Figure 1.8 Security identification for a complex ticker change history.

The symbol object does not change despite the ticker rename, remaining a consistent connection to the company represented. The `id` property of the Symbol object stores the encoded hash, which can be useful for transport (encoding to JavaScript Object Notation [JSON]) or storing on disk (Figure 1.9).

Using the symbol object for trading ensures the referenced entity remains the same through ticker renames and corporate mergers. When serialized to a string, it looks something like `SPY R735QTJ8XC9X` and is stored in the symbol.id property. This two-part string is a base64 encoded set of data that contains the IPO ticker, security type, a date, the option strike, right, and the market the asset is listed in. The `market` property

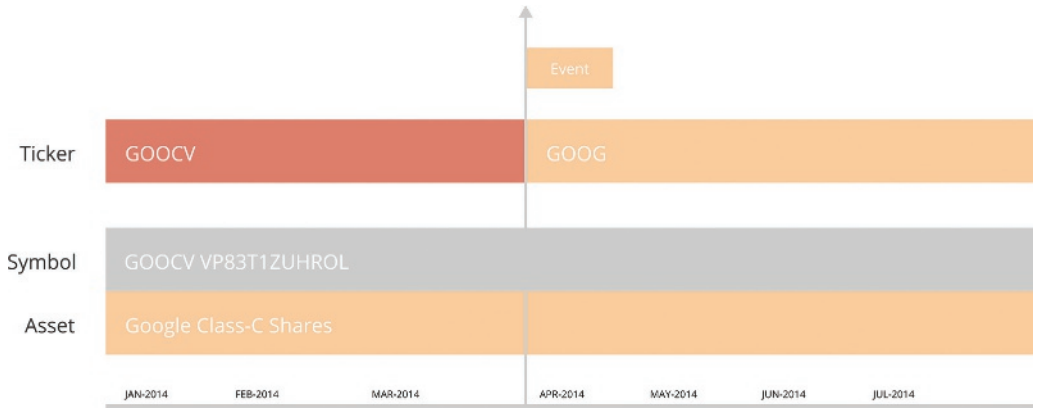

Figure 1.9 Security identification for a simple ticker change history.

distinguishes between tickers that have the same string value but represent different underlying assets—for example, BTCUSD is listed on Coinbase and on Kraken, but they have different prices and are treated as different venues because they're not easily sold on the other exchange.

Assets and Derivatives

For the purposes of the book, the following chapter is focused on five asset classes that are used in the later examples. We'll explore adding the assets to your algorithm and how to access the data used in later examples.

Quant modeling is easiest in liquid markets where well-formatted and maintained data exist. The majority of QuantConnect price data is supplied by Algoseek (algoseek.com). Established in 2015, Algoseek provides excellent, point-in-time data recorded directly from the exchanges and low latency real-time feeds. It offers scalable pricing plans that include leasing data, so quants can access large datasets for lower prices.

Most assets are listed and eventually delisted. In QuantConnect assets are called *securities* (qnt.co/book-securities) and are stored in a central securities collection. Most asset classes share common properties, such as market hours, margin requirements, profit-loss accounting, and security properties like contract multipliers.

```
self.securities["SPY"]                    # SPY security from collection
self.securities["SPY"].price              # Current price of SPY
self.securities["SPY"].symbol             # SPY symbol object
self.securities["SPY"].holdings.quantity  # Shares of SPY held
```

US Equities

Founded in 1817, the US Equity market is the largest and most liquid market in the world. It represents half of the world's market capitalization and has roughly 9,000 companies listed.

Events in a company's life cycle, such as the IPO, regular and special dividends, stock splits and reverse splits, mergers and acquisitions, and so on, are called corporate actions. These events impact the value of a company and are important factors to understand. In the following chapters, we will explore examples of how AI trained on corporate actions may be deployed in trading strategies.

Corporate financial performance data, such as revenue, costs, and profitability reported on financial statements–balance sheets, income statements, cash flow statements, and earnings forecast data—are a company's fundamental data. Fundamental data providers dutifully record company quarterly filings to the SEC. Some standardize (or harmonize) reported items into adjusted numbers that can be compared across industries and markets with different accounting conventions or practices. From financial statement data, financial analysts formulate financial ratios such as the price-earnings (PE) ratio and free-cash flow to asset ratio as common metrics to compare different companies' valuations. In the following chapters, we will explore examples of applying machine-learning techniques to predict asset prices or formulate trading strategies based on these fundamental ratios.

In QuantConnect you can easily fetch fundamental data for individual companies and develop a stock screen based on fundamental data.

```
self.add_universe(self._filter)  # Add a filtered equity data universe

def _filter(self, fundamental):  # Select symbols based on pe_ratio
    return [c.symbol for c in fundamental if c.valuation_ratios.pe_ratio < 10]
```

Some ratios used later in the book are described here:

- PE Ratio—Close price of the asset divided by earnings per share.
- Revenue Growth—Growth of the company revenue in percent from income statements.
- Free Cash Flow Percent—Free Cash Flow as a percentage of operating cash flow.
- Dividend Payout Ratio—Ratio of the dividend payments to net earnings.

US Equity Corporate Events

Over a company life cycle, many of the same corporate events occur, starting with the IPO, secondary offerings, cash or stock dividends, stock splits and reverse splits, ticker and company name changes, mergers and acquisitions, and finally delistings. In QuantConnect, these are modeled as follows:

Splits—Companies can divide or consolidate shares to make them more accessible to market participants when the price changes dramatically (Figure 1.10). As company valuation grows, the price per share can exceed the purchasing power of retail investors. A classic example is Berkshire Hathaway Class-A stock, which has never split and is worth more than $200,000 per share today. This leads to lower transaction volume and a potentially less efficient market. Similarly, as an asset price falls, it encounters

pressure from the exchanges to keep the price above $1 to remain listed—by consolidating shares (e.g., swapping two for one) in a **reverse split,** a company can double its list price to remain compliant with exchange rules.

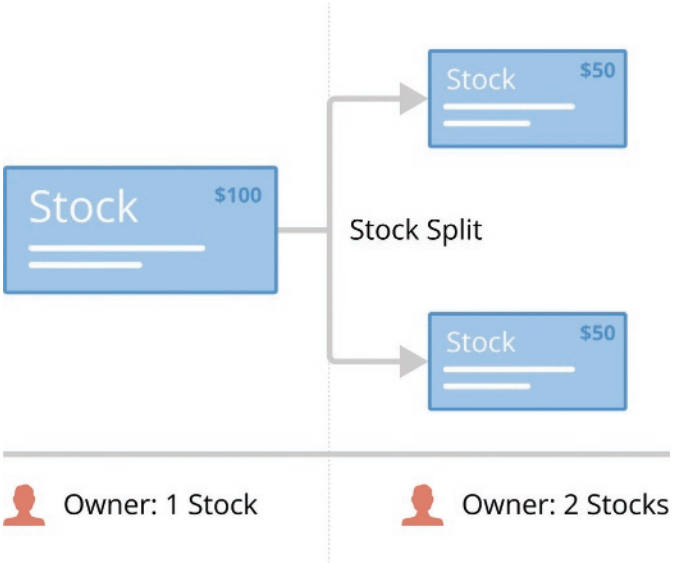

Figure 1.10 Stock value and number of shares owned before and after a split event.

In QuantConnect, splits are events passed to the split event handler that can trigger an algorithmic response for trading or incorporate in the data series for research:

```
def on_splits(self, splits):                # Dedicated splits event handler
    split = splits.get(self._symbol)

def on_data(self, slice:                    # Splits in on_data events
    split = slice.splits.get(self._symbol)
```

Splits change the price of an asset instantly, so care is needed to reset algorithm states, such as indicators that might depend on continuous prices.

Dividends—As companies pass profitability, some elect to issue dividends to shareholders. These dividends are usually paid quarterly, but ultimately, the distribution amount and frequency are up to the company's chief financial officer and the board. Dividends are quoted in dollars per share and distributed as cash to your brokerage. In QuantConnect, these dividends are passed into the dividend event handler and can be used to elicit an algorithmic response:

```
def on_data(self, slice):
    dividend = slice.dividends.get(self._symbol)

def on_dividends(self, dividends):
    dividend = dividends.get(self._symbol)
```

You can request historical data for both splits and dividends. This is helpful to manually calculate an asset's dividend yield or forecast dividend trends.

```
# Fetch all dividends for symbols paid in the last year
history = self.history[Dividend](symbols, timedelta(365))
```

The value received from paid dividends can then be modeled as a payment in cash, or applied to the price of the asset through a process called data normalization.

Data Normalization—To accurately compare two securities, we need to adjust the historical prices to account for price splits and dividend payments in a process called data normalization. These adjustments use a formula to change historical prices to reflect their value growth as accurately and smoothly as possible. There are issues with data normalization, which we won't go into in this book. Still, as a first pass, the adjusted price data does a good job of simplifying the research process by incorporating splits and dividends. For more realistic accounting of the impact of dividends and splits on your investment strategies, QuantConnect supports a *raw* mode, which applies the dividends as cash to your portfolio and lets you decide how and when to reinvest them.

In QuantConnect, you select the data normalization mode while adding the assets to your strategy:

```
# No adjustment of historical prices, and dividends paid as cash
self.add_equity("AAPL", data_normalization_mode=DataNormalizationMode.RAW)

# Full split and dividend adjustment of historical prices
self.add_equity("AAPL",
data_normalization_mode=DataNormalizationMode.ADJUSTED)

# Only adjusting for splits, and paying dividends as cash
self.add_equity("AAPL",
data_normalization_mode=DataNormalizationMode.SPLIT_ADJUSTED)
```

Security Changes—IPO, Renames, Spin-offs, Mergers, and Delistings—A company's regular activity generates many modeling challenges for a quant. Although these corporate events do not impact price, they track the listing status on the exchange. Handling them is important to ensure your portfolio's value is continuous through the change. We'll briefly review some of these next and show how they're handed in QuantConnect.

IPO—When a company is ready, it can apply to be publicly listed on an exchange. Once approved it will be formally listed in an IPO. When any asset is added to your algorithm, it generates a securities-changed event with the details of the new listing.

```
def on_securities_changed(self, changes):
    for security in changes.added_securities:
        Pass
```

Renames—Companies often rebrand to reflect their new business lines or to swap for a better market tickers. These ticker changes are handled in symbol-changed events, which provides the new and old ticker to your strategy. Swapping to the new ticker is not required, as QuantConnect uses the Symbol object to track assets.

```
def on_symbol_changed_events(self, changes):
    for symbol, change in changes.items():
        self.log(f"Change: { change.old_symbol} -> {change.new_symbol}")
```

Spin-offs and Mergers—Companies occasionally acquire smaller companies, engulfing the smaller company in the transaction. This is modeled as a simple delisting of the smaller company. Mergers are similar but can result in a new ticker; this is modeled as a delisting and a rename.

Delistings—Most asset classes have some form of delisting. For equities, this is the removal of the company from the public markets. The event is captured in the delistings event handler. In addition, it will show up in the securities changed event as a removed security.

```
def on_delistings(self, delistings):          # Asset delisting events
    delisting = delistings.get(self._symbol)

def on_securities_changed(self, changes):     # Removal from selected screen
    for security in changes.removed_securities:
        self.log(f'Security {security.Symbol} removed from universe')
```

US Equity Options

An option is a contract stipulating the *right* but not obligation to purchase an asset at a specific predetermined price (called the "strike price") on a specific date in the future (called "expiration date" or "expiry"). They are classified as a derivative—where the option's theoretical price strongly depends on the underlying asset. At the option expiration, in-the-money options are exchanged for shares of the underlying security.

Following are key properties of US equity options:

- Underlying Asset—This is the underlying equity for the contract, accessed with the `underlying` property.
- Option Right—Calls give the holder the right to buy stock at a specific price, while puts give the holder the right to sell the stock.
- Strike Price—Strike is the contractual price for the option and is important for determining the option's value and how it will be exercised at expiration.
- Expiration Date—The last date the options can be traded or exercised. American options may be exercised at any time and are settled by physical delivery of the underlying stock.

- Contract Multiplier—In equity options, each option contract represents 100 shares of the underlying assets. This contract multiplier should be multiplied to the listed prices to compute actual "invoice" prices and payout of the option contract.

Options data are disseminated by the Options Price Reporting Authority (OPRA), which aggregates the prices from the US options exchanges. Due to the regularly expiring contracts and strike prices, options generate orders of a magnitude more price data than equities. On a given day, roughly 1.5 million option contracts are available on approximately 4,000 of the most liquid companies. This generates hundreds of terabytes of financial data and can be challenging to process and analyze.

In QuantConnect, options data can be added individually or filtered from a universe of assets. Each underlying asset has hundreds of option contracts available, forming a universe based on each asset.

```
option = self.add_option("SPY")  # Requesting a universe of options on SPY
option.set_filter( … )           # Filter the option contracts returned

self.add_option_contract(contract_symbol)  # Specific option contract

def on_data(self, slice):                  # Using option chain in on_data
    chain = slice.option_chains.get(option.symbol)
```

When the underlying stock price is higher (lower) than the strike price of a call (put) option, the option is said to be in the money (ITM). That is, if exercised, the option will lead to positive cash flow to the buyer of the option; conversely when the underlying stock price is lower (higher) than the strike price of a call (put) option, the option is said to be out-of-the-money (OTM). OTM options can be used as an insurance policy to hedge against unlikely events.

The majority of US equity options are settled physically—through the transfer of shares according to the terms stipulated in the option contract. Although US equity options are mostly "American" options, meaning these options can be exercised before the expiration date, it is unlikely that a call option will be exercised early, as Option Pricing Theory predicts that the value of the call option before expiry should always be higher than the intrinsic value, which is the value if you exercise the option. To lock-in a gain for an ITM call option, the correct action is to "sell to close" the call option position. If you hold the ITM call option to expiry, your broker will automatically exercise on your behalf, and you will receive the underlying shares and pay for them with cash at the strike price. Of course, in this case, since the strike price is below the market price of the underlying, you may turn around and sell the shares for an immediate gain (mind the tax consequences if you sell).

On the other hand, American put options can be and often do get exercised early. If you sell a put option, the buyer may exercise their right to sell their shares to you before expiry. In this case, you will be "assigned" to buy and receive the shares above market price, incurring a loss. As an insurer, you are responsible for paying the claim. If the ITM put is held till expiration, then the put will be exercised automatically by the broker of the buyer

of the put. The "assignment" process for early exercise and exercise expiry is managed by the options exchange and facilitated by your broker on a first in, first out (FIFO) basis.

Index Options

Options on equity indices work similarly as individual equity options except for two important differences: they are "European," that is, they can only be exercised on the expiration date, and they are cash settled, with the payout computed based on the difference between the index level and the strike level multiplied by the contract multiplier.

QuantConnect supports three index option monthly option chains, along with their weekly counterparts: VIX (VIXW), NDX (NQX), and SPX (SPXW).

NDX—NASDAQ lists regular (NDX) European options on Nasdaq-100 indices. The NDX's reference index is the level of Nasdaq-100, with a contract multiplier of 100.

SPX—Chicago Board of Option Exchange (CBOE) lists regular SPX European options on S&P500 indices. The SPX's reference index is the level of S&P500, with a contract multiplier of 100.

VIX—The CBOE lists regular VIX European options calculated using SPX option bid/ask quotes that estimates the 30-day measure of the expected volatility of the S&P500 Index. It has a contract multiplier of 100.

These indexes can be added to an algorithm as follows:

```
self._index_symbol = self.add_index('SPX').symbol    # Add index to algorithm
option = self.add_index_option(self._index_symbol)  # Add options on index
option.set_filter(-2, 2, 0, 90)                      # Filter contracts
```

Due to the cash settlement, the US Internal Revenue Service also has different tax treatments for index options and equity options. Please consult your tax advisor for details. Many brokerages require special permission to trade index options. Some do not allow trading index options due to the different tax treatments. Inquire with your broker to determine your accounts' eligibility for index option trading.

US Futures

The US futures markets were originally created for agricultural products, such as wheat, corn, and soybeans in the 1850s, and later expanded into other commodities like meat, lumber, industrial and precious metals, energy resources, and financial instruments, such as bonds, indices, and currencies. They allowed commodity producers to reduce their market risk and fix a price for their produce ahead of time, so that the producers could plan their production with relative certainty of future revenue. The Chicago Mercantile Exchange (CME) Group is the largest collection of futures exchanges in the United States, closely followed by the Intercontinental Exchange (ICE) Group.

Futures contracts have an expiry date for when the settlement is performed. Contracts are generally categorized into monthly or quarterly contracts. QuantConnect supports the 160 most liquid future products. Each product has a collection of future contracts representing their discrete expiry dates, and forms a universe:

```
future = self.add_future(Futures.Indices.SP_500_E_MINI) # Add ES future
future.set_filter(0, 90)                                 # Filter 90 days
```

For each future product ("chain"), there are many, overlapping future contracts active with different expiry dates and different prices. The future contract that tracks the underlying commodity or securities (called spot) most closely is called the front month. "Rolling" is the process of selecting the next front-month contract. When this roll occurs, the price of the front-month contract jumps, as they are two different contracts with different expiry dates.

To create a normalized price series for the front contract for analysis, a virtual continuous front contract was created that stitches many contracts together to form a price series. Note, the continuous front contract price is not the true price of the underlying contract, but a normalized price series to capture the continuous movement of the fundamental value for data analysis. There are many methods of creating this normalized series, but as it is only a representation of the price moment there is no one "correct" way. There are nine possible combinations of normalization methods available in QuantConnect, which can be specified by the `add_future` method.

In QuantConnect, this continuous front contract is referenced as the *canonical* contract. The current underlying, tradable, contract is called the *mapped* asset. This can be accessed as follows:

```
future = self.add_future(Futures.Indices.SP_500_E_MINI)   # Canonical asset
adjusted_price = self.securities[future.symbol].price     # Adjusted price
raw_price = self.securities[future.mapped].price          # Raw price
```

When routing trades, you need to use the underlying mapped assets as the continuous contract is not tradable.

```
self.market_order(future.mapped, 1)       # Buy 1 contract of mapped asset
```

In live trading when the front contract rolls, there will be a discontinuity in the continuous price of mapped assets. Therefore, any indicators built on the continuous "canonical" prices should be reviewed and reset to adjust for the jump in the price of the mapped asset after a roll. After settlement, the exchange will delist the contract, triggering a delisting event in QuantConnect.

Most modern futures markets settle profit and loss daily in cash, so there is not a big cash payment at expiry. But the majority of commodities and financial futures are still settled in kind, requiring delivery of physical assets. Therefore, speculative traders, who normally do not intend to take or make delivery, should close or roll their futures

position ahead of expiry. Many brokerages have an automatic liquidation policy in place when physical-delivery futures are near expiry. This eliminates the risk of delivery that financial brokers do not support. Such liquidations can have unintended and likely negative consequences on your investment portfolio, so traders are advised to consult their brokers for an early liquidation date in order to close or roll their futures positions ahead of forced liquidation.

Cryptocurrency

In the last decade, cryptocurrency has exploded as a new global asset class. As a decentralized store of value and business ecosystem, it had wide appeal. Early adopters were well rewarded with enormous gains in value.

Trading is split between centralized exchanges and decentralized exchanges (DEX). Most exchanges were built by technologists with modern application program interfaces (APIs) but limited knowledge of the financial infrastructure underpinning traditional financial markets. As such, they operate in a similar way to mid-century US exchanges—disconnected and independent before the SIP connected them with a nationwide feed. Most rely on private market makers to provide liquidity, and other funds who perform inter-exchange arbitrage to balance asset prices between exchanges.

However, the crypto-trading infrastructure is rapidly catching up to those of traditional financial markets. The first crypto exchanges were launched with spot-price trading on cash accounts only. Now there are hundreds of exchanges offering margin trading, and trading on crypto derivatives similar to continuous futures or options.

As they have different prices and no centralized feed, crypto exchanges are considered different markets. They can be added with the add crypto API:

```
# requesting price feed for both exchanges by market
coinbase_btcusd = self.add_crypto("BTCUSD", market=Market.COINBASE).symbol
kraken_btcusd = self.add_crypto("BTCUSD", market=Market.KRAKEN).symbol
```

As each exchange has different supported margins, assets, and order types, it's important to set the brokerage model for the relevant exchange. Here you can also set the account type (cash or margin).

```
self.set_brokerage_model(BrokerageName.KRAKEN, AccountType.MARGIN)
```