PART 1

Data Analysis

Chapter 2: Financial Data Structures, 23
Chapter 3: Labeling, 43
Chapter 4: Sample Weights, 59
Chapter 5: Fractionally Differentiated Features, 75

CHAPTER 2

Financial Data Structures

2.1 MOTIVATION

In this chapter we will learn how to work with unstructured financial data, and from that to derive a structured dataset amenable to ML algorithms. In general, you do not want to consume someone else's processed dataset, as the likely outcome will be that you discover what someone else already knows or will figure out soon. Ideally your starting point is a collection of unstructured, raw data that you are going to process in a way that will lead to informative features.

2.2 ESSENTIAL TYPES OF FINANCIAL DATA

Financial data comes in many shapes and forms. Table 2.1 shows the four essential types of financial data, ordered from left to right in terms of increasing diversity. Next, we will discuss their different natures and applications.

2.2.1 Fundamental Data

Fundamental data encompasses information that can be found in regulatory filings and business analytics. It is mostly accounting data, reported quarterly. A particular aspect of this data is that it is reported with a lapse. You must confirm exactly when each data point was released, so that your analysis uses that information only after it was publicly available. A common beginner's error is to assume that this data was published at the end of the reporting period. That is never the case.

For example, fundamental data published by Bloomberg is indexed by the last date included in the report, which precedes the date of the release (often by 1.5 months). In other words, Bloomberg is assigning those values to a date when they were not known. You could not believe how many papers are published every year using misaligned

TABLE 2.1 The Four Essential Types of Financial Data

Fundamental Data	Market Data	Analytics	Alternative Data
• Assets • Liabilities • Sales • Costs/earnings • Macro variables • . . .	• Price/yield/implied volatility • Volume • Dividend/coupons • Open interest • Quotes/cancellations • Aggressor side • . . .	• Analyst recommendations • Credit ratings • Earnings expectations • News sentiment • . . .	• Satellite/CCTV images • Google searches • Twitter/chats • Metadata • . . .

fundamental data, especially in the factor-investing literature. Once you align the data correctly, a substantial number of findings in those papers cannot be reproduced.

A second aspect of fundamental data is that it is often backfilled or reinstated. "Backfilling" means that missing data is assigned a value, even if those values were unknown at that time. A "reinstated value" is a corrected value that amends an incorrect initial release. A company may issue multiple corrections for a past quarter's results long after the first publication, and data vendors may overwrite the initial values with their corrections. The problem is, the corrected values were not known on that first release date. Some data vendors circumvent this problem by storing multiple release dates and values for each variable. For example, we typically have three values for a single quarterly GDP release: the original released value and two monthly revisions. Still, it is very common to find studies that use the final released value and assign it to the time of the first release, or even to the last day in the reporting period. We will revisit this mistake, and its implications, when we discuss backtesting errors in Chapter 11.

Fundamental data is extremely regularized and low frequency. Being so accessible to the marketplace, it is rather unlikely that there is much value left to be exploited. Still, it may be useful in combination with other data types.

2.2.2 Market Data

Market data includes all trading activity that takes place in an exchange (like CME) or trading venue (like MarketAxess). Ideally, your data provider has given you a raw feed, with all sorts of unstructured information, like FIX messages that allow you to fully reconstruct the trading book, or the full collection of BWIC (bids wanted in competition) responses. Every market participant leaves a characteristic footprint in the trading records, and with enough patience, you will find a way to anticipate a competitor's next move. For example, TWAP algorithms leave a very particular footprint that is used by predatory algorithms to front-run their end-of-day trading (usually hedging) activity (Easley, López de Prado, and O'Hara [2011]). Human GUI traders often trade in round lots, and you can use this fact to estimate what percentage of the volume is coming from them at a given point in time, then associate it with a particular market behavior.

One appealing aspect of FIX data is that it is not trivial to process, unlike fundamental data. It is also very abundant, with over 10 TB being generated on a daily basis. That makes it a more interesting dataset for strategy research.

2.2.3 Analytics

You could think of analytics as derivative data, based on an original source, which could be fundamental, market, alternative, or even a collection of other analytics. What characterizes analytics is not the content of the information, but that it is not readily available from an original source, and that it has been processed for you in a particular way. Investment banks and research firms sell valuable information that results from in-depth analyses of companies' business models, activities, competition, outlook, etc. Some specialized firms sell statistics derived from alternative data, for example, the sentiment extracted from news reports and social media.

A positive aspect of analytics is that the signal has been extracted for you from a raw source. The negative aspects are that analytics may be costly, the methodology used in their production may be biased or opaque, and you will not be the sole consumer.

2.2.4 Alternative Data

Kolanovic and Krishnamachari [2017] differentiate among alternative data produced by individuals (social media, news, web searches, etc.), business processes (transactions, corporate data, government agencies, etc.), and sensors (satellites, geolocation, weather, CCTV, etc.). Some popular satellite image or video feeds include monitoring of tankers, tunnel traffic activity, or parking lot occupancies.

What truly characterizes alternative data is that it is primary information, that is, information that has not made it to the other sources. Before Exxon Mobile reported increased earnings, before its market price shot up, before analysts wrote their commentary of their latest filings, before all of that, there were movements of tankers and drillers and pipeline traffic. They happened months before those activities were reflected in the other data types. Two problematic aspects of alternative data are their cost and privacy concerns. All that spy craft is expensive, and the surveilled company may object, not to mention bystanders.

Alternative data offers the opportunity to work with truly unique, hard-to-process datasets. Remember, data that is hard to store, manipulate, and operate is always the most promising. You will recognize that a dataset *may be* useful if it annoys your data infrastructure team. Perhaps your competitors did not try to use it for logistic reasons, gave up midway, or processed it incorrectly.

2.3 BARS

In order to apply ML algorithms on your unstructured data, we need to parse it, extract valuable information from it, and store those extractions in a regularized format. Most ML algorithms assume a table representation of the extracted data.

Finance practitioners often refer to those tables' rows as "bars." We can distinguish between two categories of bar methods: (1) standard bar methods, which are common in the literature, and (2) more advanced, information-driven methods, which sophisticated practitioners use although they cannot be found (yet) in journal articles. In this section, we will discuss how to form those bars.

2.3.1 Standard Bars

Some bar construction methods are very popular in the financial industry, to the point that most data vendors' APIs offer several of them. The purpose of these methods is to transform a series of observations that arrive at irregular frequency (often referred to as "inhomogeneous series") into a homogeneous series derived from regular sampling.

2.3.1.1 Time Bars

Time bars are obtained by sampling information at fixed time intervals, e.g., once every minute. The information collected usually includes:

- Timestamp
- Volume-weighted average price (VWAP)
- Open (i.e., first) price
- Close (i.e., last) price
- High price
- Low price
- Volume traded, etc.

Although time bars are perhaps the most popular among practitioners and academics, they should be avoided for two reasons. First, markets do not process information at a constant time interval. The hour following the open is much more active than the hour around noon (or the hour around midnight in the case of futures). As biological beings, it makes sense for humans to organize their day according to the sunlight cycle. But today's markets are operated by algorithms that trade with loose human supervision, for which CPU processing cycles are much more relevant than chronological intervals (Easley, López de Prado, and O'Hara [2011]). This means that time bars oversample information during low-activity periods and undersample information during high-activity periods. Second, time-sampled series often exhibit poor statistical properties, like serial correlation, heteroscedasticity, and non-normality of returns (Easley, López de Prado, and O'Hara [2012]). GARCH models were developed, in part, to deal with the heteroscedasticity associated with incorrect sampling. As we will see next, forming bars as a subordinated process of trading activity avoids this problem in the first place.

2.3.1.2 Tick Bars

The idea behind tick bars is straightforward: The sample variables listed earlier (timestamp, VWAP, open price, etc.) will be extracted each time a pre-defined number

of transactions takes place, e.g., 1,000 ticks. This allows us to synchronize sampling with a proxy of information arrival (the speed at which ticks are originated).

Mandelbrot and Taylor [1967] were among the first to realize that sampling as a function of the number of transactions exhibited desirable statistical properties: "Price changes over a fixed number of transactions may have a Gaussian distribution. Price changes over a fixed time period may follow a stable Paretian distribution, whose variance is infinite. Since the number of transactions in any time period is random, the above statements are not necessarily in disagreement."

Ever since Mandelbrot and Taylor's paper, multiple studies have confirmed that sampling as a function of trading activity allows us to achieve returns closer to IID Normal (see Ané and Geman [2000]). This is important, because many statistical methods rely on the assumption that observations are drawn from an IID Gaussian process. Intuitively, we can only draw inference from a random variable that is invariant, and tick bars allow for better inference than time bars.

When constructing tick bars, you need to be aware of outliers. Many exchanges carry out an auction at the open and an auction at the close. This means that for a period of time, the order book accumulates bids and offers without matching them. When the auction concludes, a large trade is published at the clearing price, for an outsized amount. This auction trade could be the equivalent of thousands of ticks, even though it is reported as one tick.

2.3.1.3 Volume Bars

One problem with tick bars is that order fragmentation introduces some arbitrariness in the number of ticks. For example, suppose that there is one order sitting on the offer, for a size of 10. If we buy 10 lots, our one order will be recorded as one tick. If instead on the offer there are 10 orders of size 1, our one buy will be recorded as 10 separate transactions. In addition, matching engine protocols can further split one fill into multiple artificial partial fills, as a matter of operational convenience.

Volume bars circumvent that problem by sampling every time a pre-defined amount of the security's units (shares, futures contracts, etc.) have been exchanged. For example, we could sample prices every time a futures contract exchanges 1,000 units, regardless of the number of ticks involved.

It is hard to imagine these days, but back in the 1960s vendors rarely published volume data, as customers were mostly concerned with tick prices. After volume started to be reported as well, Clark [1973] realized that sampling returns by volume achieved even better statistical properties (i.e., closer to an IID Gaussian distribution) than sampling by tick bars. Another reason to prefer volume bars over time bars or tick bars is that several market microstructure theories study the interaction between prices and volume. Sampling as a function of one of these variables is a convenient artifact for these analyses, as we will find out in Chapter 19.

2.3.1.4 Dollar Bars

Dollar bars are formed by sampling an observation every time a pre-defined market value is exchanged. Of course, the reference to dollars is meant to apply to the

currency in which the security is denominated, but nobody refers to euro bars, pound bars, or yen bars (although gold bars would make for a fun pun).

Let me illustrate the rationale behind dollar bars with a couple of examples. First, suppose that we wish to analyze a stock that has exhibited an appreciation of 100% over a certain period of time. Selling $1,000 worth of that stock at the end of the period requires trading half the number of shares it took to buy $1,000 worth of that stock at the beginning. In other words, the number of shares traded is a function of the actual value exchanged. Therefore, it makes sense sampling bars in terms of dollar value exchanged, rather than ticks or volume, particularly when the analysis involves significant price fluctuations. This point can be verified empirically. If you compute tick bars and volume bars on E-mini S&P 500 futures for a given bar size, the number of bars per day will vary wildly over the years. That range and speed of variation will be reduced once you compute the number of dollar bars per day over the years, for a constant bar size. Figure 2.1 plots the exponentially weighted average number of bars per day when we apply a fixed bar size on tick, volume, and dollar sampling methods.

A second argument that makes dollar bars more interesting than time, tick, or volume bars is that the number of outstanding shares often changes multiple times over the course of a security's life, as a result of corporate actions. Even after adjusting for splits and reverse splits, there are other actions that will impact the amount of ticks and volumes, like issuing new shares or buying back existing shares (a very common practice since the Great Recession of 2008). Dollar bars tend to be robust in the face of those actions. Still, you may want to sample dollar bars where the size of the bar is not kept constant over time. Instead, the bar size could be adjusted dynamically as a function of the free-floating market capitalization of a company (in the case of stocks), or the outstanding amount of issued debt (in the case of fixed-income securities).

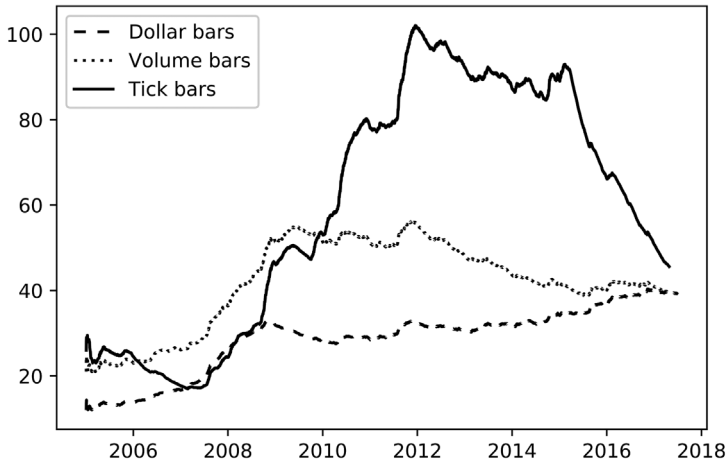

FIGURE 2.1 Average daily frequency of tick, volume, and dollar bars

2.3.2 Information-Driven Bars

The purpose of information-driven bars is to sample more frequently when new information arrives to the market. In this context, the word "information" is used in a market microstructural sense. As we will see in Chapter 19, market microstructure theories confer special importance to the persistence of imbalanced signed volumes, as that phenomenon is associated with the presence of informed traders. By synchronizing sampling with the arrival of informed traders, we may be able to make decisions before prices reach a new equilibrium level. In this section we will explore how to use various indices of information arrival to sample bars.

2.3.2.1 Tick Imbalance Bars

Consider a sequence of ticks $\{(p_t, v_t)\}_{t=1,\ldots,T}$, where p_t is the price associated with tick t and v_t is the volume associated with tick t. The so-called tick rule defines a sequence $\{b_t\}_{t=1,\ldots,T}$ where

$$b_t = \begin{cases} b_{t-1} & \text{if } \Delta p_t = 0 \\ \dfrac{|\Delta p_t|}{\Delta p_t} & \text{if } \Delta p_t \neq 0 \end{cases}$$

with $b_t \in \{-1, 1\}$, and the boundary condition b_0 is set to match the terminal value b_T from the immediately preceding bar. The idea behind tick imbalance bars (TIBs) is to sample bars whenever tick imbalances exceed our expectations. We wish to determine the tick index, T, such that the accumulation of signed ticks (signed according to the tick rule) exceeds a given threshold. Next, let us discuss the procedure to determine T.

First, we define the tick imbalance at time T as

$$\theta_T = \sum_{t=1}^{T} b_t$$

Second, we compute the expected value of θ_T at the beginning of the bar, $\mathrm{E}_0[\theta_T] = \mathrm{E}_0[T](\mathrm{P}[b_t = 1] - \mathrm{P}[b_t = -1])$, where $\mathrm{E}_0[T]$ is the expected size of the tick bar, $\mathrm{P}[b_t = 1]$ is the unconditional probability that a tick is classified as a buy, and $\mathrm{P}[b_t = -1]$ is the unconditional probability that a tick is classified as a sell. Since $\mathrm{P}[b_t = 1] + \mathrm{P}[b_t = -1] = 1$, then $\mathrm{E}_0[\theta_T] = \mathrm{E}_0[T](2\mathrm{P}[b_t = 1] - 1)$. In practice, we can estimate $\mathrm{E}_0[T]$ as an exponentially weighted moving average of T values from prior bars, and $(2\mathrm{P}[b_t = 1] - 1)$ as an exponentially weighted moving average of b_t values from prior bars.

Third, we define a tick imbalance bar (TIB) as a T^*-contiguous subset of ticks such that the following condition is met:

$$T^* = \arg\min_T \left\{ \left|\theta_T\right| \geq \mathrm{E}_0\left[T\right] \left| 2\mathrm{P}\left[b_t = 1\right] - 1 \right| \right\}$$

where the size of the expected imbalance is implied by $|2\mathrm{P}[b_t = 1] - 1|$. When θ_T is more imbalanced than expected, a low T will satisfy these conditions. Accordingly, TIBs are produced more frequently under the presence of informed trading (asymmetric information that triggers one-side trading). In fact, we can understand TIBs as buckets of trades containing equal amounts of information (regardless of the volumes, prices, or ticks traded).

2.3.2.2 Volume/Dollar Imbalance Bars

The idea behind volume imbalance bars (VIBs) and dollar imbalance bars (DIBs) is to extend the concept of tick imbalance bars (TIBs). We would like to sample bars when volume or dollar imbalances diverge from our expectations. Based on the same notions of tick rule and boundary condition b_0 as we discussed for TIBs, we will define a procedure to determine the index of the next sample, T.

First, we define the imbalance at time T as

$$\theta_T = \sum_{t=1}^{T} b_t v_t$$

where v_t may represent either the number of securities traded (VIB) or the dollar amount exchanged (DIB). Your choice of v_t is what determines whether you are sampling according to the former or the latter.

Second, we compute the expected value of θ_T at the beginning of the bar

$$\begin{aligned}\mathrm{E}_0[\theta_T] = \mathrm{E}_0\left[\sum_{t|b_t=1}^{T} v_t\right] - \mathrm{E}_0\left[\sum_{t|b_t=-1}^{T} v_t\right] = \mathrm{E}_0[T](\mathrm{P}[b_t = 1]\mathrm{E}_0[v_t|b_t = 1] \\ -\mathrm{P}[b_t = -1]\mathrm{E}_0[v_t|b_t = -1])\end{aligned}$$

Let us denote $v^+ = \mathrm{P}[b_t = 1]\mathrm{E}_0[v_t|b_t = 1]$, $v^- = \mathrm{P}[b_t = -1]\mathrm{E}_0[v_t|b_t = -1]$, so that $\mathrm{E}_0[T]^{-1}E_0[\sum_t v_t] = \mathrm{E}_0[v_t] = v^+ + v^-$. You can think of v^+ and v^- as decomposing the initial expectation of v_t into the component contributed by buys and the component contributed by sells. Then

$$\mathrm{E}_0[\theta_T] = \mathrm{E}_0[T](v^+ - v^-) = \mathrm{E}_0[T](2v^+ - \mathrm{E}_0[v_t])$$

In practice, we can estimate $\mathrm{E}_0[T]$ as an exponentially weighted moving average of T values from prior bars, and $(2v^+ - \mathrm{E}_0[v_t])$ as an exponentially weighted moving average of $b_t v_t$ values from prior bars.

Third, we define VIB or DIB as a T^*-contiguous subset of ticks such that the following condition is met:

$$T^* = \arg\min_T\{|\theta_T| \geq \mathrm{E}_0[T]|2v^+ - \mathrm{E}_0[v_t]|\}$$

where the size of the expected imbalance is implied by $|2v^+ - \mathrm{E}_0[v_t]|$. When θ_T is more imbalanced than expected, a low T will satisfy these conditions. This is the information-based analogue of volume and dollar bars, and like its predecessors, it addresses the same concerns regarding tick fragmentation and outliers. Furthermore, it also addresses the issue of corporate actions, because the above procedure does not rely on a constant bar size. Instead, the bar size is adjusted dynamically.

2.3.2.3 Tick Runs Bars

TIBs, VIBs, and DIBs monitor order flow imbalance, as measured in terms of ticks, volumes, and dollar values exchanged. Large traders will sweep the order book, use iceberg orders, or slice a parent order into multiple children, all of which leave a trace of runs in the $\{b_t\}_{t=1,\ldots,T}$ sequence. For this reason, it can be useful to monitor the *sequence* of buys in the overall volume, and take samples when that sequence diverges from our expectations.

First, we define the length of the current run as

$$\theta_T = \max\left\{\sum_{t|b_t=1}^{T} b_t, -\sum_{t|b_t=-1}^{T} b_t\right\}$$

Second, we compute the expected value of θ_T at the beginning of the bar

$$\mathrm{E}_0[\theta_T] = \mathrm{E}_0[T]\max\{\mathrm{P}[b_t = 1], 1 - \mathrm{P}[b_t = 1]\}$$

In practice, we can estimate $\mathrm{E}_0[T]$ as an exponentially weighted moving average of T values from prior bars, and $\mathrm{P}[b_t = 1]$ as an exponentially weighted moving average of the proportion of buy ticks from prior bars.

Third, we define a tick runs bar (TRB) as a T^*-contiguous subset of ticks such that the following condition is met:

$$T^* = \arg\min_T\{\theta_T \geq \mathrm{E}_0[T]\max\{\mathrm{P}[b_t = 1],1 - \mathrm{P}[b_t = 1]\}\}$$

where the expected count of ticks from runs is implied by $\max\{\mathrm{P}[b_t = 1], 1 - \mathrm{P}[b_t = -1]\}$. When θ_T exhibits more runs than expected, a low T will satisfy these conditions. Note that in this definition of runs we allow for sequence breaks. That is, instead of measuring the length of the longest sequence, we count the number of ticks of each side, without offsetting them (no imbalance). In the context of forming bars, this turns out to be a more useful definition than measuring sequence lengths.

2.3.2.4 Volume/Dollar Runs Bars

Volume runs bars (VRBs) and dollar runs bars (DRBs) extend the above definition of runs to volumes and dollars exchanged, respectively. The intuition is that we wish to sample bars whenever the volumes or dollars traded by one side exceed our expectation for a bar. Following our customary nomenclature for the tick rule, we need to determine the index T of the last observation in the bar.

First, we define the volumes or dollars associated with a run as

$$\theta_T = \max\left\{\sum_{t|b_t=1}^{T} b_t v_t, -\sum_{t|b_t=-1}^{T} b_t v_t\right\}$$

where v_t may represent either number of securities traded (VRB) or dollar amount exchanged (DRB). Your choice of v_t is what determines whether you are sampling according to the former or the latter.

Second, we compute the expected value of θ_T at the beginning of the bar,

$$\mathrm{E}_0[\theta_T] = \mathrm{E}_0[T]\max\{\mathrm{P}[b_t = 1]\mathrm{E}_0[v_t|b_t = 1], (1 - \mathrm{P}[b_t = 1])\mathrm{E}_0[v_t|b_t = -1]\}$$

In practice, we can estimate $\mathrm{E}_0[T]$ as an exponentially weighted moving average of T values from prior bars, $\mathrm{P}[b_t = 1]$ as an exponentially weighted moving average of the proportion of buy ticks from prior bars, $\mathrm{E}_0[v_t|b_t = 1]$ as an exponentially weighted moving average of the buy volumes from prior bars, and $\mathrm{E}_0[v_t|b_t = -1]$ as an exponentially weighted moving average of the sell volumes from prior bars.

Third, we define a volume runs bar (VRB) as a T^*-contiguous subset of ticks such that the following condition is met:

$$\begin{aligned} T^* = \arg\min_T\{\theta_T \geq \mathrm{E}_0[T]\max\{\mathrm{P}[b_t = 1]\mathrm{E}_0[v_t|b_t = 1], \\ (1 - \mathrm{P}[b_t = 1])\mathrm{E}_0[v_t|b_t = -1]\}\} \end{aligned}$$

where the expected volume from runs is implied by $\max\{\mathrm{P}[b_t = 1]\mathrm{E}_0[v_t|b_t = 1], (1 - \mathrm{P}[b_t = 1])\mathrm{E}_0[v_t|b_t = -1]\}$. When θ_T exhibits more runs than expected, or the volume from runs is greater than expected, a low T will satisfy these conditions.

2.4 DEALING WITH MULTI-PRODUCT SERIES

Sometimes we are interested in modelling a time series of instruments, where the weights need to be dynamically adjusted over time. Other times we must deal with products that pay irregular coupons or dividends, or that are subject to corporate actions. Events that alter the nature of the time series under study need to be treated properly, or we will inadvertently introduce a structural break that will mislead our research efforts (more on this in Chapter 17). This problem appears in many guises: when we model spreads with changing weights, or baskets of securities where dividends/coupons must be reinvested, or baskets that must be rebalanced, or when an index's constituents are changed, or when we must replace an expired/matured contract/security with another, etc.

Futures are a case in point. In my experience, people struggle unnecessarily when manipulating futures, mainly because they do not know how to handle the roll well. The same can be said of strategies based on spreads of futures, or baskets of stocks or bonds. In the next section, I'll show you one way to model a basket of securities as if it was a single cash product. I call it the "ETF trick" because the goal is to transform any complex multi-product dataset into a single dataset that resembles a total-return ETF. Why is this useful? Because your code can always assume that you only trade cashlike products (non-expiring cash instruments), regardless of the complexity and composition of the underlying series.

2.4.1 The ETF Trick

Suppose we wish to develop a strategy that trades a spread of futures. A few nuisances arise from dealing with a spread rather than an outright instrument. First, the spread is characterized by a vector of weights that changes over time. As a result, the spread itself may converge even if prices do not change. When that happens, a model trading that series will be misled to believe that PnL (the net mark-to-market value of profits and losses) has resulted from that weight-induced convergence. Second, spreads can acquire negative values, because they do not represent a price. This can often be problematic, as most models assume positive prices. Third, trading times will not align exactly for all constituents, so the spread is not always tradeable at the last levels published, or with zero latency risk. Also, execution costs must be considered, like crossing the bid-ask spread.

One way to avoid these issues is to produce a time series that reflects the value of $1 invested in a spread. Changes in the series will reflect changes in PnL, the series will be strictly positive (at worst, infinitesimal), and the implementation shortfall will be taken into account. This will be the series used to model, generate signals, and trade, as if it were an ETF.

Suppose that we are given a history of bars, as derived from any of the methods explained in Section 2.3. These bars contain the following columns:

- $o_{i,t}$ is the raw open price of instrument $i = 1, \ldots, I$ at bar $t = 1, \ldots, T$.
- $p_{i,t}$ is the raw close price of instrument $i = 1, \ldots, I$ at bar $t = 1, \ldots, T$.
- $\varphi_{i,t}$ is the USD value of one point of instrument $i = 1, \ldots, I$ at bar $t = 1, \ldots, T$. This includes foreign exchange rate.
- $v_{i,t}$ is the volume of instrument $i = 1, \ldots, I$ at bar $t = 1, \ldots, T$.
- $d_{i,t}$ is the carry, dividend, or coupon paid by instrument i at bar t. This variable can also be used to charge margin costs, or costs of funding.

where all instruments $i = 1, \ldots, I$ were tradeable at bar $t = 1, \ldots, T$. In other words, even if some instruments were not tradeable over the entirety of the time interval $[t-1, t]$, at least they were tradeable at times $t-1$ and t (markets were open and able to execute orders at those instants). For a basket of futures characterized by an

allocations vector ω_t rebalanced (or rolled) on bars $B \subseteq \{1, \ldots, T\}$, the \$1 investment value $\{K_t\}$ is derived as

$$h_{i,t} = \begin{cases} \dfrac{\omega_{i,t} K_t}{o_{i,t+1} \varphi_{i,t} \sum_{i=1}^{I} |\omega_{i,t}|} & \text{if } t \in B \\ h_{i,t-1} & \text{otherwise} \end{cases}$$

$$\delta_{i,t} = \begin{cases} p_{i,t} - o_{i,t} & \text{if } (t-1) \in B \\ \Delta p_{i,t} & \text{otherwise} \end{cases}$$

$$K_t = K_{t-1} + \sum_{i=1}^{I} h_{i,t-1} \varphi_{i,t} \left(\delta_{i,t} + d_{i,t} \right)$$

and $K_0 = 1$ in the initial AUM. Variable $h_{i,t}$ represents the holdings (number of securities or contracts) of instrument i at time t. Variable $\delta_{i,t}$ is the change of market value between $t-1$ and t for instrument i. Note that profits or losses are being reinvested whenever $t \in B$, hence preventing the negative prices. Dividends $d_{i,t}$ are already embedded in K_t, so there is no need for the strategy to know about them. The purpose of $\omega_{i,t} \left(\sum_{i=1}^{I} |\omega_{i,t}| \right)^{-1}$ in $h_{i,t}$ is to de-lever the allocations. For series of futures, we may not know $p_{i,t}$ of the new contract at a roll time t, so we use $o_{i,t+1}$ as the closest in time.

Let τ_i be the transaction cost associated with trading \$1 of instrument i, e.g., $\tau_i = 1E-4$ (one basis point). There are three additional variables that the strategy needs to know for every observed bar t:

1. **Rebalance costs:** The variable cost $\{c_t\}$ associated with the allocation rebalance is $c_t = \sum_{i=1}^{I} (|h_{i,t-1}| p_{i,t} + |h_{i,t}| o_{i,t+1}) \varphi_{i,t} \tau_i, \forall t \in B$. We do not embed c_t in K_t, or shorting the spread will generate fictitious profits when the allocation is rebalanced. In your code, you can treat $\{c_t\}$ as a (negative) dividend.
2. **Bid-ask spread:** The cost $\{\tilde{c}_t\}$ of buying or selling one unit of this virtual ETF is $\tilde{c}_t = \sum_{i=1}^{I} |h_{i,t-1}| p_{i,t} \varphi_{i,t} \tau_i$. When a unit is bought or sold, the strategy must charge this cost $\tilde{c}_t$, which is the equivalent to crossing the bid-ask spread of this virtual ETF.
3. **Volume:** The volume traded $\{v_t\}$ is determined by the least active member in the basket. Let $v_{i,t}$ be the volume traded by instrument i over bar t. The number of tradeable basket units is $v_t = \min_i \left\{ \frac{v_{i,t}}{|h_{i,t-1}|} \right\}$.

Transaction costs functions are not necessarily linear, and those non-linear costs can be simulated by the strategy based on the above information. Thanks to the ETF trick, we can model a basket of futures (or a single futures) as if it was a single non-expiring cash product.

2.4.2 PCA Weights

The interested reader will find many practical ways of computing hedging weights in López de Prado and Leinweber [2012] and Bailey and López de Prado [2012]. For the sake of completeness, let us review one way to derive the vector $\{\omega_t\}$ used in the previous section. Consider an IID multivariate Gaussian process characterized by a vector of means μ, of size *Nx1*, and a covariance matrix V, of size *NxN*. This stochastic process describes an invariant random variable, like the returns of stocks, the changes in yield of bonds, or changes in options' volatilities, for a portfolio of N instruments. We would like to compute the vector of allocations ω that conforms to a particular distribution of risks across V's principal components.

First, we perform a spectral decomposition, $VW = W\Lambda$, where the columns in W are reordered so that the elements of Λ's diagonal are sorted in descending order. Second, given a vector of allocations ω, we can compute the portfolio's risk as $\sigma^2 = \omega' V\omega = \omega' W\Lambda W'\omega = \beta'\Lambda\beta = (\Lambda^{1/2}\beta)'(\Lambda^{1/2}\beta)$, where β represents the projection of ω on the orthogonal basis. Third, Λ is a diagonal matrix, thus $\sigma^2 = \sum_{n=1}^{N} \beta_n^2 \Lambda_{n,n}$, and the risk attributed to the nth component is $R_n = \beta_n^2 \Lambda_{n,n}\sigma^{-2} = [W'\omega]_n^2 \Lambda_{n,n}\sigma^{-2}$, with $R'1_N = 1$, and 1_N is a vector of N ones. You can interpret $\{R_n\}_{n=1,\ldots,N}$ as the distribution of risks across the orthogonal components.

Fourth, we would like to compute a vector ω that delivers a user-defined risk distribution R. From earlier steps, $\beta = \left\{\pm\sigma\sqrt{\frac{R_n}{\Lambda_{n,n}}}\right\}_{n=1,\ldots,N}$, which represents the allocation in the new (orthogonal) basis. Fifth, the allocation in the old basis is given by $\omega = W\beta$. Re-scaling ω merely re-scales σ, hence keeping the risk distribution constant. Figure 2.2 illustrates the contribution to risk per principal component for an

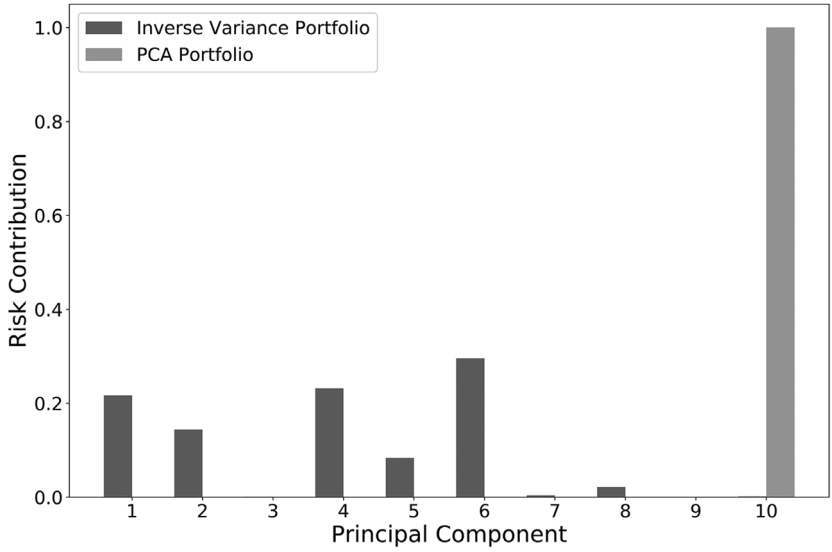

FIGURE 2.2 Contribution to risk per principal component

inverse variance allocation. Almost all principal components contribute risk, including those with highest variance (components 1 and 2). In contrast, for the PCA portfolio, only the component with lowest variance contributes risk.

Snippet 2.1 implements this method, where the user-defined risk distribution R is passed through argument `riskDist` (optional None). If `riskDist` is None, the code will assume all risk must be allocated to the principal component with smallest eigenvalue, and the weights will be the last eigenvector re-scaled to match σ (`riskTarget`).

SNIPPET 2.1 PCA WEIGHTS FROM A RISK DISTRIBUTION R

```
def pcaWeights(cov,riskDist=None,riskTarget=1.):
    # Following the riskAlloc distribution, match riskTarget
    eVal,eVec=np.linalg.eigh(cov) # must be Hermitian
    indices=eVal.argsort()[::-1] # arguments for sorting eVal desc
    eVal,eVec=eVal[indices],eVec[:,indices]
    if riskDist is None:
        riskDist=np.zeros(cov.shape[0])
        riskDist[-1]=1.
    loads=riskTarget*(riskDist/eVal)**.5
    wghts=np.dot(eVec,np.reshape(loads,(-1,1)))
    #ctr=(loads/riskTarget)**2*eVal # verify riskDist
    return wghts
```

2.4.3 Single Future Roll

The ETF trick can handle the rolls of a single futures contract, as a particular case of a 1-legged spread. However, when dealing with a single futures contract, an equivalent and more direct approach is to form a time series of cumulative roll gaps, and detract that gaps series from the price series. Snippet 2.2 shows a possible implementation of this logic, using a series of tick bars downloaded from Bloomberg and stored in a HDF5 table. The meaning of the Bloomberg fields is as follows:

- `FUT_CUR_GEN_TICKER`: It identifies the contract associated with that price. Its value changes with every roll.
- `PX_OPEN`: The open price associated with that bar.
- `PX_LAST`: The close price associated with the bar.
- `VWAP`: The volume-weighted average price associated with that bar.

The argument `matchEnd` in function `rollGaps` determines whether the futures series should be rolled forward (`matchEnd=False`) or backward (`matchEnd=True`). In a forward roll, the price at the start of the rolled series

matches the price at the start of the raw series. In a backward roll, the price at the end of the rolled series matches the price at the end of the raw series.

SNIPPET 2.2 FORM A GAPS SERIES, DETRACT IT FROM PRICES

```
def getRolledSeries(pathIn,key):
    series=pd.read_hdf(pathIn,key='bars/ES_10k')
    series['Time']=pd.to_datetime(series['Time'],format='%Y%m%d%H%M%S%f')
    series=series.set_index('Time')
    gaps=rollGaps(series)
    for fld in ['Close','VWAP']:series[fld]-=gaps
    return series
#-----------------------------------------
def rollGaps(series,dictio={'Instrument':'FUT_CUR_GEN_TICKER','Open':'PX_OPEN', \
    'Close':'PX_LAST'},matchEnd=True):
    # Compute gaps at each roll, between previous close and next open
    rollDates=series[dictio['Instrument']].drop_duplicates(keep='first').index
    gaps=series[dictio['Close']]*0
    iloc=list(series.index)
    iloc=[iloc.index(i)-1 for i in rollDates] # index of days prior to roll
    gaps.loc[rollDates[1:]]=series[dictio['Open']].loc[rollDates[1:]]- \
        series[dictio['Close']].iloc[iloc[1:]].values
    gaps=gaps.cumsum()
    if matchEnd:gaps-=gaps.iloc[-1] # roll backward
    return gaps
```

Rolled prices are used for simulating PnL and portfolio mark-to-market values. However, raw prices should still be used to size positions and determine capital consumption. Keep in mind, rolled prices can indeed become negative, particularly in futures contracts that sold off while in contango. To see this, run Snippet 2.2 on a series of Cotton #2 futures or Natural Gas futures.

In general, we wish to work with non-negative rolled series, in which case we can derive the price series of a $1 investment as follows: (1) Compute a time series of rolled futures prices, (2) compute the return (r) as rolled price change divided by the previous raw price, and (3) form a price series using those returns (i.e., `(1+r).cumprod()`). Snippet 2.3 illustrates this logic.

SNIPPET 2.3 NON-NEGATIVE ROLLED PRICE SERIES

```
raw=pd.read_csv(filePath,index_col=0,parse_dates=True)
gaps=rollGaps(raw,dictio={'Instrument':'Symbol','Open':'Open','Close':'Close'})
rolled=raw.copy(deep=True)
for fld in ['Open','Close']:rolled[fld]-=gaps
rolled['Returns']=rolled['Close'].diff()/raw['Close'].shift(1)
rolled['rPrices']=(1+rolled['Returns']).cumprod()
```

2.5 SAMPLING FEATURES

So far we have learned how to produce a continuous, homogeneous, and structured dataset from a collection of unstructured financial data. Although you could attempt to apply an ML algorithm on such a dataset, in general that would not be a good idea, for a couple of reasons. First, several ML algorithms do not scale well with sample size (e.g., SVMs). Second, ML algorithms achieve highest accuracy when they attempt to learn from relevant examples. Suppose that you wish to predict whether the next 5% absolute return will be positive (a 5% rally) or negative (a 5% sell-off). At any random time, the accuracy of such a prediction will be low. However, if we ask a classifier to predict the sign of the next 5% absolute return after certain catalytic conditions, we are more likely to find informative features that will help us achieve a more accurate prediction. In this section we discuss ways of sampling bars to produce a features matrix with relevant training examples.

2.5.1 Sampling for Reduction

As we have mentioned earlier, one reason for sampling features from a structured dataset is to reduce the amount of data used to fit the ML algorithm. This operation is also referred to as *downsampling*. This is often done by either sequential sampling at a constant step size (linspace sampling), or by sampling randomly using a uniform distribution (uniform sampling).

The main advantage of linspace sampling is its simplicity. The disadvantages are that the step size is arbitrary, and that outcomes may vary depending on the seed bar. Uniform sampling addresses these concerns by drawing samples uniformly across the entire set of bars. Still, both methods suffer the criticism that the sample does not necessarily contain the subset of most relevant observations in terms of their predictive power or informational content.

2.5.2 Event-Based Sampling

Portfolio managers typically place a bet after some event takes place, such as a structural break (Chapter 17), an extracted signal (Chapter 18), or microstructural phenomena (Chapter 19). These events could be associated with the release of some macroeconomic statistics, a spike in volatility, a significant departure in a spread away from its equilibrium level, etc. We can characterize an event as significant, and let the ML algorithm learn whether there is an accurate prediction function under those circumstances. Perhaps the answer is no, in which case we would redefine what constitutes an event, or try again with alternative features. For illustration purposes, let us discuss one useful event-based sampling method.

2.5.2.1 The CUSUM Filter

The CUSUM filter is a quality-control method, designed to detect a shift in the mean value of a measured quantity away from a target value. Consider IID

observations $\{y_t\}_{t=1,\ldots,T}$ arising from a locally stationary process. We define the cumulative sums

$$S_t = \max\left\{0, S_{t-1} + y_t - \mathrm{E}_{t-1}\left[y_t\right]\right\}$$

with boundary condition $S_0 = 0$. This procedure would recommend an action at the first t satisfying $S_t \geq h$, for some threshold h (the filter size). Note that $S_t = 0$ whenever $y_t \leq \mathrm{E}_{t-1}[y_t] - S_{t-1}$. This zero floor means that we will skip some downward deviations that otherwise would make S_t negative. The reason is, the filter is set up to identify a sequence of upside divergences from any reset level zero. In particular, the threshold is activated when

$$S_t \geq h \Leftrightarrow \exists \tau \in [1, t] \left| \sum_{i=\tau}^{t} \left(y_i - \mathrm{E}_{i-1}\left[y_t\right]\right) \geq h \right.$$

This concept of run-ups can be extended to include run-downs, giving us a symmetric CUSUM filter:

$$S_t^+ = \max\left\{0, S_{t-1}^+ + y_t - \mathrm{E}_{t-1}\left[y_t\right]\right\},\ S_0^+ = 0$$

$$S_t^- = \min\left\{0, S_{t-1}^- + y_t - \mathrm{E}_{t-1}\left[y_t\right]\right\},\ S_0^- = 0$$

$$S_t = \max\left\{S_t^+, -S_t^-\right\}$$

Lam and Yam [1997] propose an investment strategy whereby alternating buy-sell signals are generated when an absolute return h is observed relative to a prior high or low. Those authors demonstrate that such strategy is equivalent to the so-called "filter trading strategy" studied by Fama and Blume [1966]. Our use of the CUSUM filter is different: We will sample a bar t if and only if $S_t \geq h$, at which point S_t is reset. Snippet 2.4 shows an implementation of the symmetric CUSUM filter, where $\mathrm{E}_{t-1}[y_t] = y_{t-1}$.

SNIPPET 2.4 THE SYMMETRIC CUSUM FILTER

```
def getTEvents(gRaw,h):
    tEvents,sPos,sNeg=[],0,0
    diff=gRaw.diff()
    for i in diff.index[1:]:
        sPos,sNeg=max(0,sPos+diff.loc[i]),min(0,sNeg+diff.loc[i])
        if sNeg<-h:
            sNeg=0;tEvents.append(i)
        elif sPos>h:
            sPos=0;tEvents.append(i)
    return pd.DatetimeIndex(tEvents)
```

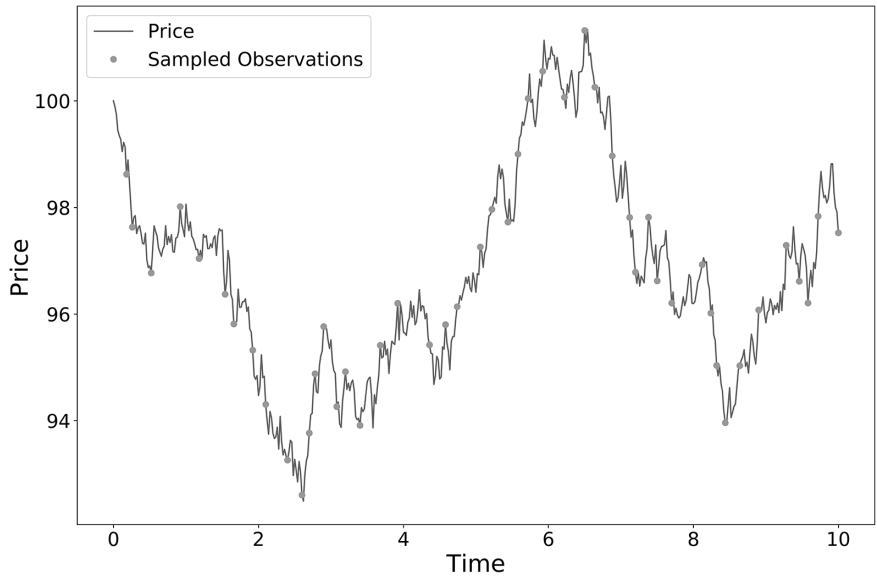

FIGURE 2.3 CUSUM sampling of a price series

The function `getTEvents` receives two arguments: the raw time series we wish to filter (`gRaw`) and the threshold, `h`. One practical aspect that makes CUSUM filters appealing is that multiple events are not triggered by `gRaw` hovering around a threshold level, which is a flaw suffered by popular market signals such as Bollinger bands. It will require a full run of length `h` for `gRaw` to trigger an event. Figure 2.3 illustrates the samples taken by a CUSUM filter on a price series.

Variable S_t could be based on any of the features we will discuss in Chapters 17–19, like structural break statistics, entropy, or market microstructure measurements. For example, we could declare an event whenever SADF departs sufficiently from a previous reset level (to be defined in Chapter 17). Once we have obtained this subset of event-driven bars, we will let the ML algorithm determine whether the occurrence of such events constitutes actionable intelligence.

EXERCISES

2.1 On a series of E-mini S&P 500 futures tick data:

- **(a)** Form tick, volume, and dollar bars. Use the ETF trick to deal with the roll.
- **(b)** Count the number of bars produced by tick, volume, and dollar bars on a weekly basis. Plot a time series of that bar count. What bar type produces the most stable weekly count? Why?
- **(c)** Compute the serial correlation of returns for the three bar types. What bar method has the lowest serial correlation?

(d) Partition the bar series into monthly subsets. Compute the variance of returns for every subset of every bar type. Compute the variance of those variances. What method exhibits the smallest variance of variances?

(e) Apply the Jarque-Bera normality test on returns from the three bar types. What method achieves the lowest test statistic?

2.2 On a series of E-mini S&P 500 futures tick data, compute dollar bars and dollar imbalance bars. What bar type exhibits greater serial correlation? Why?

2.3 On dollar bar series of E-mini S&P 500 futures and Eurostoxx 50 futures:

(a) Apply Section 2.4.2 to compute the $\{\hat{\omega}_t\}$ vector used by the ETF trick. (Hint: You will need FX values for EUR/USD at the roll dates.)

(b) Derive the time series of the S&P 500/Eurostoxx 50 spread.

(c) Confirm that the series is stationary, with an ADF test.

2.4 Form E-mini S&P 500 futures dollar bars:

(a) Compute Bollinger bands of width 5% around a rolling moving average. Count how many times prices cross the bands out (from within the bands to outside the bands).

(b) Now sample those bars using a CUSUM filter, where $\{y_t\}$ are returns and $h = 0.05$. How many samples do you get?

(c) Compute the rolling standard deviation of the two-sampled series. Which one is least heteroscedastic? What is the reason for these results?

2.5 Using the bars from exercise 4:

(a) Sample bars using the CUSUM filter, where $\{y_t\}$ are absolute returns and $h = 0.05$.

(b) Compute the rolling standard deviation of the sampled bars.

(c) Compare this result with the results from exercise 4. What procedure delivered the least heteroscedastic sample? Why?

REFERENCES

Ané, T. and H. Geman (2000): "Order flow, transaction clock and normality of asset returns." *Journal of Finance*, Vol. 55, pp. 2259–2284.

Bailey, David H., and M. López de Prado (2012): "Balanced baskets: A new approach to trading and hedging risks." *Journal of Investment Strategies (Risk Journals)*, Vol. 1, No. 4 (Fall), pp. 21–62.

Clark, P. K. (1973): "A subordinated stochastic process model with finite variance for speculative prices." *Econometrica*, Vol. 41, pp. 135–155.

Easley, D., M. López de Prado, and M. O'Hara (2011): "The volume clock: Insights into the high frequency paradigm." *Journal of Portfolio Management*, Vol. 37, No. 2, pp. 118–128.

Easley, D., M. López de Prado, and M. O'Hara (2012): "Flow toxicity and liquidity in a high frequency world." *Review of Financial Studies*, Vol. 25, No. 5, pp. 1457–1493.

Fama, E. and M. Blume (1966): "Filter rules and stock market trading." *Journal of Business*, Vol. 40, pp. 226–241.

Kolanovic, M. and R. Krishnamachari (2017): "Big data and AI strategies: Machine learning and alternative data approach to investing." White paper, JP Morgan, Quantitative and Derivatives Strategy. May 18.

Lam, K. and H. Yam (1997): "CUSUM techniques for technical trading in financial markets." *Financial Engineering and the Japanese Markets*, Vol. 4, pp. 257–274.

López de Prado, M. and D. Leinweber (2012): "Advances in cointegration and subset correlation hedging methods." *Journal of Investment Strategies (Risk Journals)*, Vol. 1, No. 2 (Spring), pp. 67–115.

Mandelbrot, B. and M. Taylor (1967): "On the distribution of stock price differences." *Operations Research*, Vol. 15, No. 5, pp. 1057–1062.