Financial Data Structure

What are we going to learn today?

- Essential Types of Financial Data
- Bars
  - standard bars
  - information-driven bars
- Dealing with Multi-Product Series
- Sampling Features

Essential Types of Financial Data

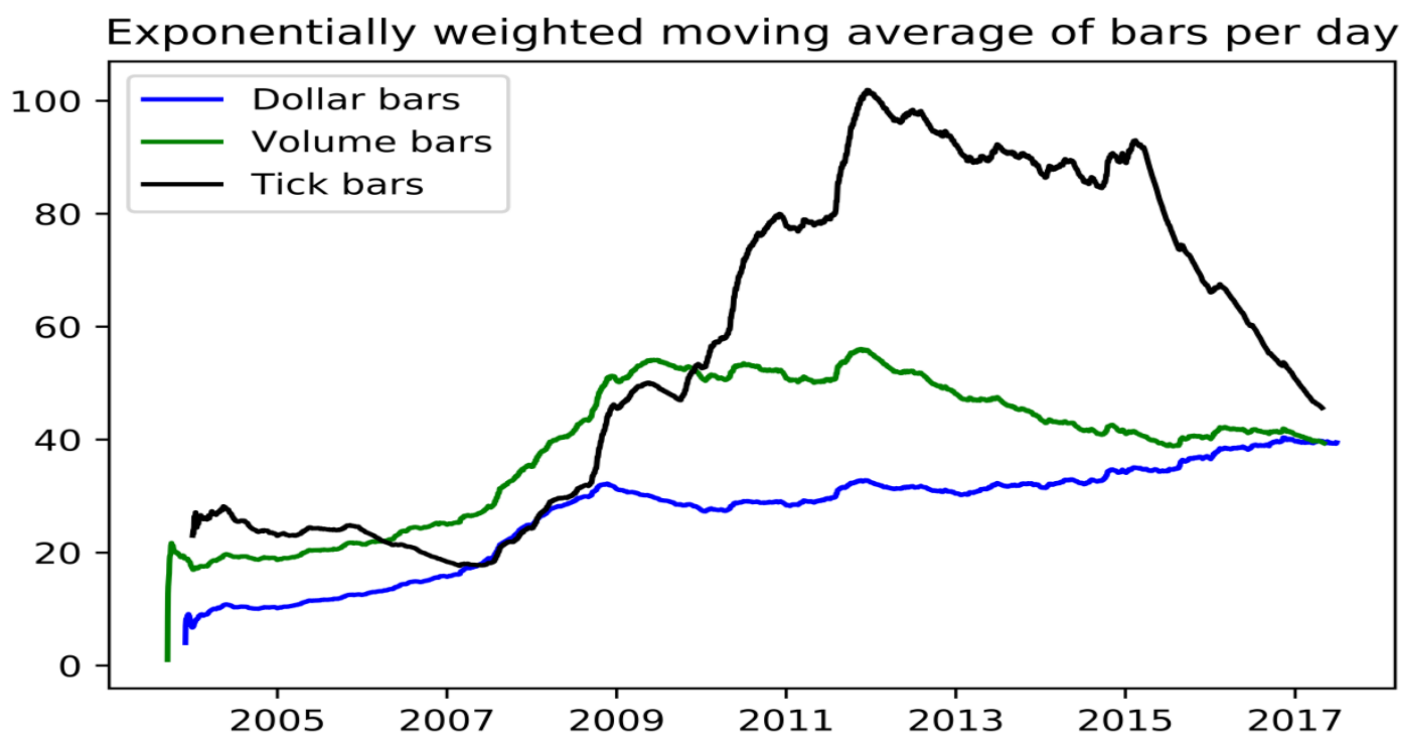
Fundamental Data	Market Data	Analytics	Alternative Data
• Assets • Liabilities • Sales • Costs/earnings • Macro variables • . . .	• Price/yield/implied volatility • Volume • Dividend/coupons • Open interest • Quotes/cancellations • Aggressor side • . . .	• Analyst recommendations • Credit ratings • Earnings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

Bars

Forming Bars

- Information does not arrive to the market at a constant entropy rate
- Sampling data in chronological intervals means that the informational content of the individual observations is far from constant
- A better approach is to sample observations as a subordinated process of the amount of information exchanged:
    - Trade bars
    - Volume bars
    - Dollar bars
    - Volatility or run bars
    - Order imbalance bars
    - Entropy bars

Sampling Frequencies



Three bar types computed on E-mini S&P 500 futures.

Tick bars tend to exhibit a wide range of sampling frequencies, for multiple microstructural reasons.

Sampling frequencies for **volume bars** are often inversely proportional to price levels.

In general, **dollar bars** tend to exhibit more stable sampling frequencies.

Dollar Imbalance Bars (1/2)

- Let's define the imbalance at time T as $\theta_T = \sum_{t=1}^{T} b_t v_t$, where $b_t \in \{-1,1\}$ is the aggressor flag, and v_t may represent either the number of securities traded or the dollar amount exchanged.
- We compute the expected value of θ_T at the beginning of the bar

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

- Let's 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.

Dollar Imbalance Bars (2/2)

- 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.
- We define a bar 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.

Multi-Product Series

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.

```
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
```

(*) For rolling baskets of futures & options, see also Section 2.4.1 (the “ETF Trick”)

Sampling Features

Ways to Do Feature Sampling

One reason for sampling features from a structured dataset is to reduce the amount of data used to fit the ML algorithm. The operation is also referred to as ***downsampling***

- Sampling for reduction
    - linspace sampling
    - uniform sampling

- Event-Based Sampling
    - macroeconomic statistics: a spike in volatility, a significant departure in a spread away from its equilibrium level

The CUMSUM 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)
```

Example of CUSUM filter