Data Structures

Advances in Financial Machine Learning

GridGain

HUDSON AND THAMES

In-Memory Computing SUMMIT | EUROPE 2019

- Cost: £525 -> Free for attendees
- Link: https://www.imcsummit.org/2019/eu/register
- Code: TomVIP
- 2019 June 3-4

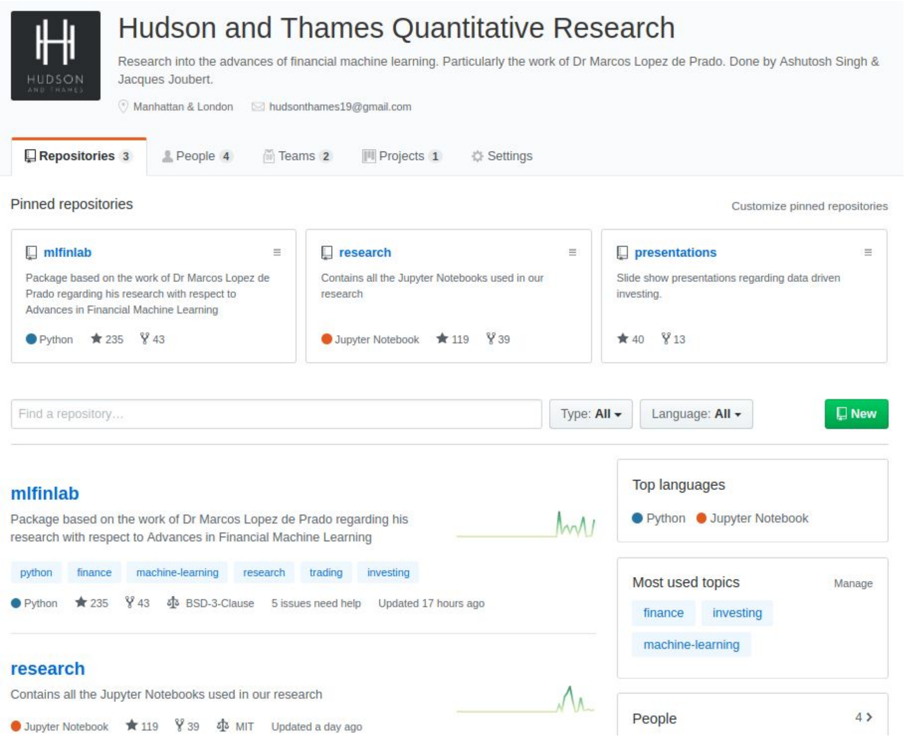
Follow us @imcsummit

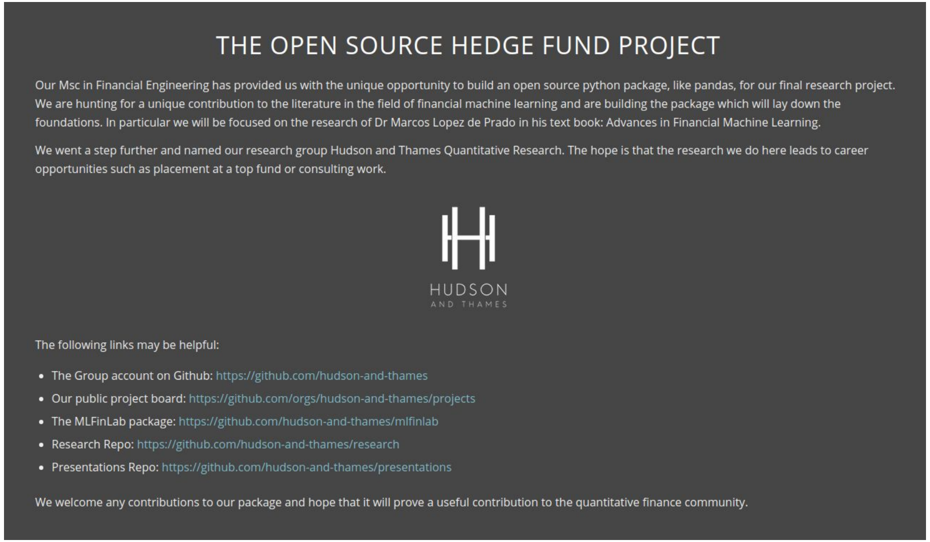
"Make your Data Science Actionable with Stream Processing." Hazelcast's Riaz Mohammed to share an architecture for… https://t.co/G9iCyeo4Bl

Overview

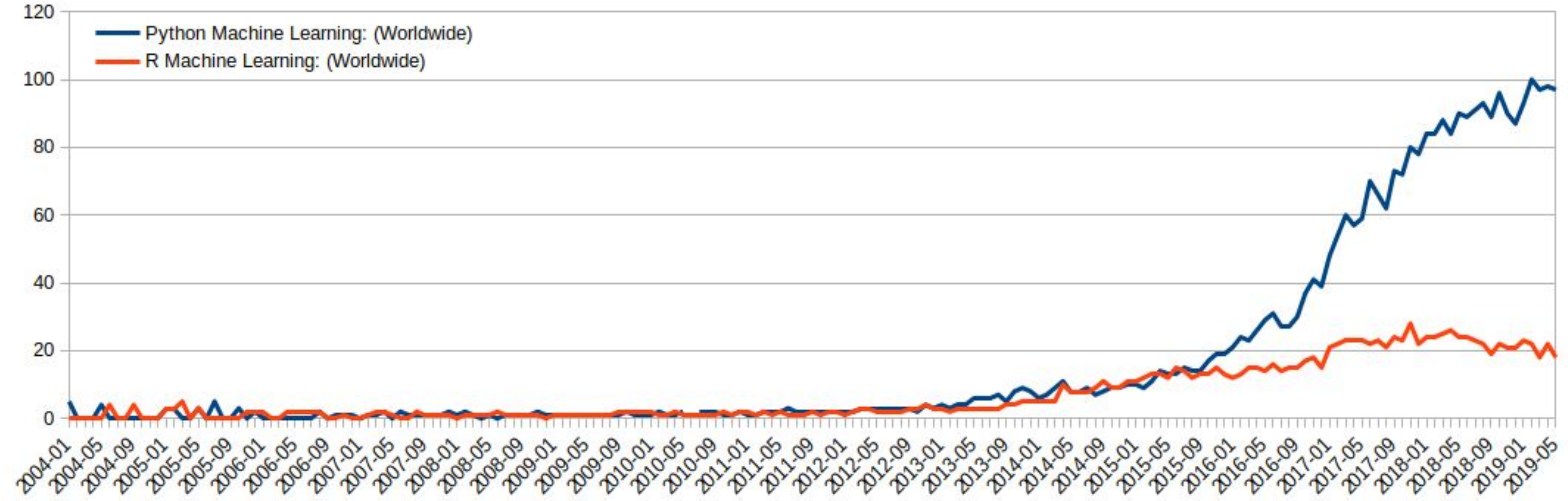
This presentation covers the various financial data structures mentioned in Advances in Financial Machine Learning and includes a getting started tutorial on using mlfinlab.

Open Source Hedge Fund Project

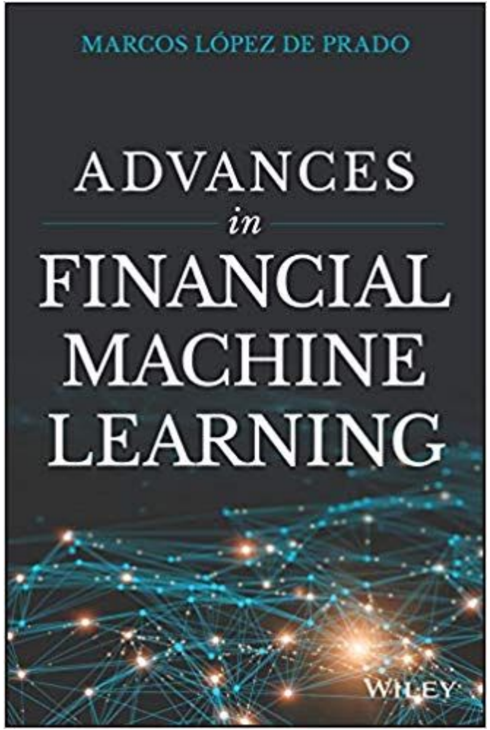




Python


Python Machine Learning: (Worldwide)
R Machine Learning: (Worldwide)
120
100
80
60
40
20
0
2004-01 2004-05 2004-09 2005-01 2005-05 2005-09 2006-01 2006-05 2006-09 2007-01 2007-05 2007-09 2008-01 2008-05 2008-09 2009-01 2009-05 2009-09 2010-01 2010-05 2010-09 2011-01 2011-05 2011-09 2012-01 2012-05 2012-09 2013-01 2013-05 2013-09 2014-01 2014-05 2014-09 2015-01 2015-05 2015-09 2016-01 2016-05 2016-09 2017-01 2017-05 2017-09 2018-01 2018-05 2018-09 2019-01 2019-05

Advances in Financial Machine Learning



www.quantresearch.org

Why Most Machine Learning Funds Fail

The 10 Reasons Most Machine Learning Funds Fail

Marcos López de Prado

Marcos López de Prado
is a research fellow at Lawrence Berkeley National Laboratory in Berkeley, CA.
lopezdeprado@lbl.gov

For almost a century, economics and finance have relied almost exclusively on the econometric toolkit to perform empirical analyses. The essential tool of econometrics is multivariate linear regression, an 18th-century technology that was already mastered by Gauss in 1794 (Stigler [1981]). Standard econometric models do not learn. It is hard to believe that something as complex as 21st-century finance could be grasped by something as simple as inverting a covariance matrix.

Every empirical science must build theories based on observation. If the statistical toolbox used to model these observations is linear regression, the researcher will fail to recognize the complexity of the data, and the theories will be awfully simplistic and useless. To this day, no one has been able to prove a theorem stating that risk premiums must be linear. Hence, reducing our analysis to linear regressions is likely a mistake. Econometrics may be a primary reason economics and finance have not experienced meaningful progress over the past 70 years (Calkin and López de Prado [2014a, 2014b]).

For centuries, medieval astronomers made observations and developed theories about celestial mechanics. These theories never considered noncircular orbits because they were deemed unholy and beneath God's plan. The prediction errors were so gross that ever more complex theories had to be devised to account for new observations. It was not until Kepler had the temerity to consider noncircular (elliptical) orbits that, all of a sudden, a much simpler general model was able to predict the position of the planets with astonishing accuracy. What if astronomers had never considered noncircular orbits? Well, what if economists finally started to consider nonlinear functions? Where is our Kepler? Finance does not have a *Principia* because no Kepler means no Newton.

In recent years, quantitative fund managers have experimented and succeeded with the use of machine learning (ML) methods. An ML algorithm learns patterns in a high-dimensional space without being specifically directed. A common misconception is that ML methods are black boxes. This is not necessarily true. When correctly used, ML models do not replace theory; they guide it. Once we understand what features are predictive of a phenomenon, we can build a theoretical explanation that can be tested on an independent dataset. Students of economics and finance would do well to enroll in ML courses rather than econometrics. Econometrics may be good enough to succeed in financial academia (for now), but succeeding in business requires ML.

At the same time, ML is no panacea. The flexibility and power of ML techniques have a dark side. When misused, ML algorithms will confuse statistical flukes

10 Reasons (Journal of Portfolio Management)

1. Working in Silos
2. Research Through Backtesting
3. **Chronological Sampling**
4. Integer Differentiation
5. Fixed-Time Horizon Labeling
6. Learning Side and Size Simultaneously
7. Weighting of Non-Independent Identically Distributed Samples
8. Cross-Validation Leakage
9. Walk-Forward (or Historical) Backtesting
10. Backtest Overfitting

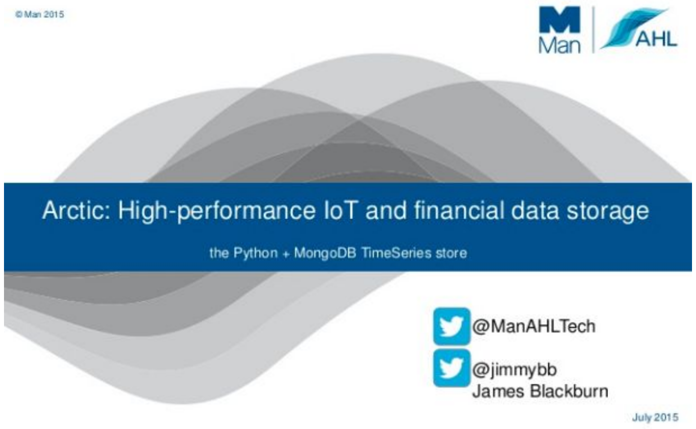
GridGain HUDSON AND THAMES

Types of Financial Data

1. Fundamental Data. (ZIP features, Fama French 3 Factor)
    a. Very low latency
    b. Easily accessible
    c. Unlikely to find unexploited value
    d. May still prove useful
2. Market Data
    a. Fix messages
    b. BWIC (Bids wanted in competition)
    c. Harder to process, lots of data.
    d. Makes for interesting strategy research
3. Analytics
    a. Created using an original source, processed.
    b. Features or data created to enhance models
    c. Sentiment signals, strategy signals
4. Alternative Data
    a. Satellite image, traffic in a mall, cars in a parking lot.

To avoid discovering what others have, your starting point should be the raw data that you are going to process in unique ways - with the goal of discovering new informative features.

“Data that is hard to store, manipulate, and operate is always the most promising”



Types of Bars

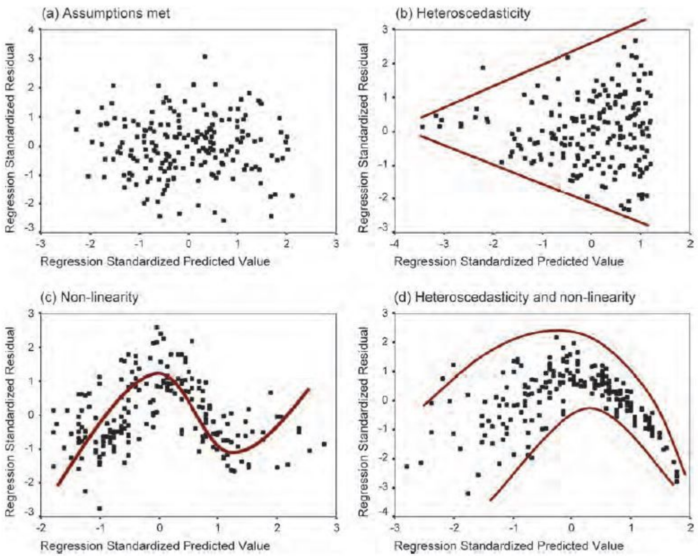
1. Standard Bars
    a. Time
    b. Tick
    c. Volume
    d. Dollar
2. Information Driven Bars
    a. Imbalance
    b. Run

date_time	open	high	low	close	volume
09/01/2013 19:32:23.387	1640.25	1642.00	1639.00	1642.00	28031
09/02/2013 01:18:21.928	1642.00	1644.00	1640.25	1643.50	28003
09/02/2013 02:50:32.992	1643.50	1646.00	1642.25	1644.75	28000
09/02/2013 04:57:09.236	1644.75	1647.25	1643.75	1646.00	28000
09/02/2013 07:04:32.076	1646.00	1648.50	1645.75	1647.50	28013

Time Bars

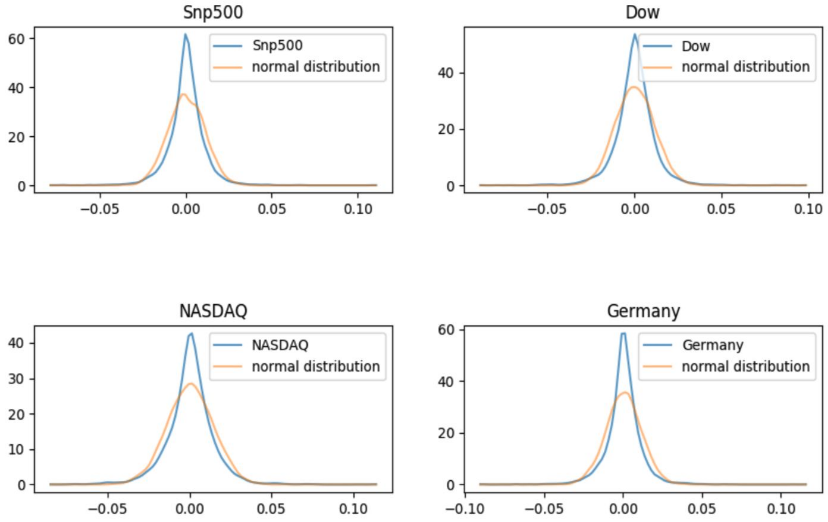
Time bars (fixed interval sampling) are the most common financial data structure, however they should be avoid for 2 reasons:

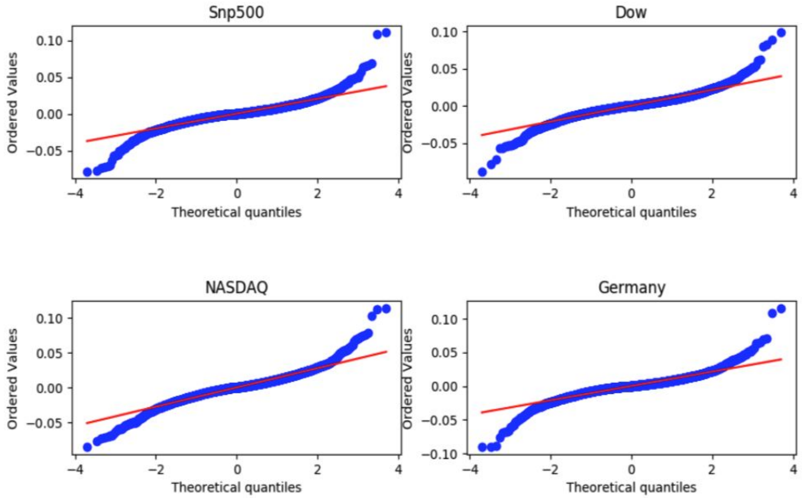
1. Markets don’t process information in fixed time intervals
   a. Over and undersampling
2. Poor statistical properties:
   a. Serial correlation
   b. Heteroscedasticity
   c. Non-normality of returns



Assumption checking

Chronological Clock: Distribution Comparison





Statistical Advantages:

The idea of a different time clock was first identified in Mandelbrot and Taylor 1967.

Multiple studies have confirmed that sampling as a function of trading activity results in returns closer to IID normal. Important because many statistical tools rely on the assumption that observations are drawn from an IID Gaussian process.

1. Removes most intra-session seasonal effects
2. Partial recovery of normality and the IID assumption
3. Addresses the problem of random and asynchronous transactions (a major concern when computing correlations on HFT data)
4. Reduces the impact of volatility clustering, because large price moves are associated with large volumes. Sampling by volume is a proxy for sampling by volatility. (less heteroskedastic)

ON THE DISTRIBUTION OF STOCK PRICE DIFFERENCES

Benoit Mandelbrot
International Business Machines, Yorktown Heights, New York
and
Howard M. Taylor
Cornell University, Ithaca, New York

(Received March 20, 1967)

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. A possible explanation is proposed by TAYLOR, and then shown by MANDELBROT to be intimately related to an earlier discussion of the specialists' function of ensuring the continuity of the market.

THERE ARE at least four schools of thought on the statistical distribution of stock price differences, or more generally, stochastic models for sequences of stock prices. In terms of number of followers, by far the most popular approach is that of the so-called 'technical analyst' phrased in terms of short term trends, support and resistance levels, technical rebounds, and so on. Rejecting this technical viewpoint, two other schools agree that sequences of prices describe a random walk, where price changes are statistically independent of previous price history, but these schools disagree in their choice of the appropriate probability distributions and/or in their choice of the appropriate 'time' parameter (the physical time—days, hours—or a randomized operational time ruled by the flow of transactions). Some authors find price changes to be normal or Gaussian,[1,2,8,14,15] while the other group find them to follow a stable Paretian law with infinite variance.[3,4,9,11] Finally, a fourth group (overlapping with the preceding two) admits the random walk as a first-order approximation but notes recognizable second-order effects.[10,12,13,16]

Basically, our point is this: the Gaussian random walk as applied to transactions is compatible with a symmetric stable Paretian random walk as applied to fixed time intervals.

New Standard Bar Types

Tick bars

- Susceptible to outliers, be careful of the auctions (Many ticks grouped into one)
- Order fragmentation: Limit order for 10 stocks but lifted with 4 separate market orders.
    - Matching engines can also split orders into artificial partially filled orders.

Dollar Bars

Solve problems of large price changes as it samples as a function of value traded. This also resolves some of the problems relating to corporate actions such as share splits.

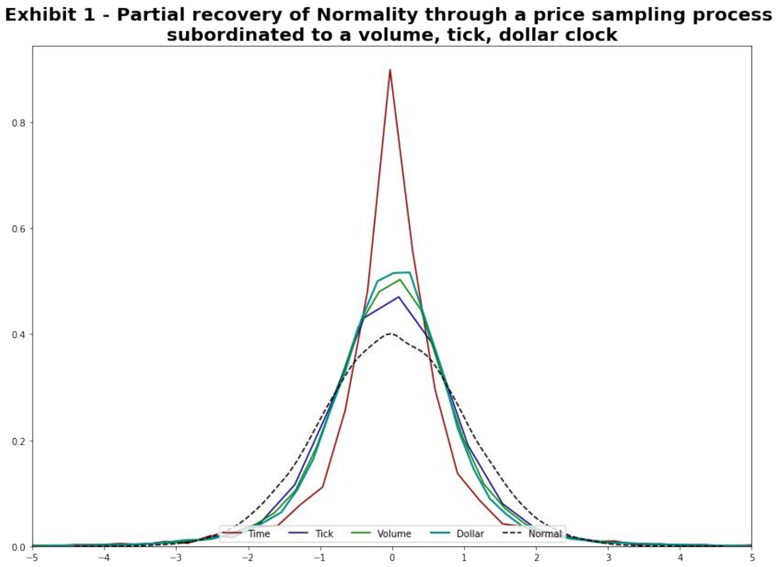
Volume Bars

Overcome the problems of tick bars by sampling every time a fixed number of units are traded.

Clark 1973, realized that sampling returns by volume had better statistical properties than sampling by ticks (Closer to a normal distribution).

One problem is if prices have very large moves then volumes will drop, If prices drop significantly then we will see an increase in volumes traded.

Better Sampling Techniques



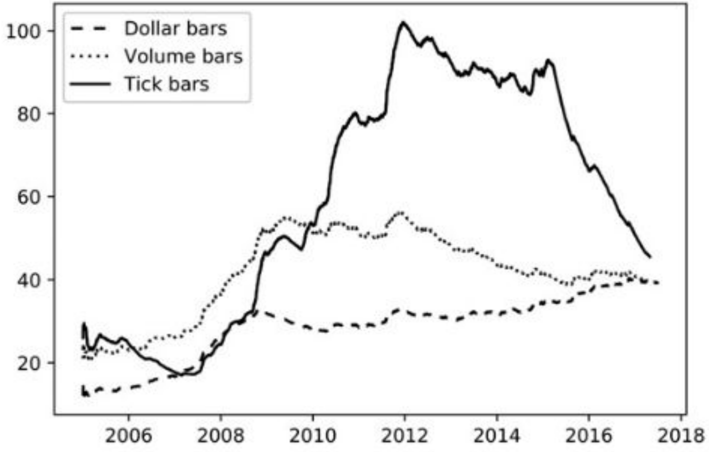


FIGURE 2.1 Average daily frequency of tick, volume, and dollar bars

Barriers to Entry

No open-source implementations to create the financial data structures:

1. Must be fast
2. Work on large files
3. Reliable (Unit Tests)

Tick data is expense. No sample data available.

- Provide 2 year sample for various financial data structures. (Not raw tick data)

README.md

Sample Data

The following folder contains 2 years sample data on S&P500 Emini Futures, for the period 2015-01-01 to 2017-01-01.

Specifically the following data structures:

- Dollar Bars: Sampled every $70'000
- Volume Bars: Sampled every 28'000 contracts
- Tick Bars: Sampled every 2'800 ticks

The following fields are available:

- Date Time
- Open
- High
- Low
- Close
- Cumulative Dollars
- Cumulative Volume
- Cumulative Ticks

Recreate Data

To create the data structures from first principles, make use of the mlfinlab package. We made use of raw tick data.

Purpose

Our hope is that the following samples will enable the community to build on the research and contribute to the open source community.

A good place to start for new users is to use the data provided to answer the questions at the back of chapter 2 of Advances in Financial Machine Learning.

Notebook Walkthrough

README.md

Notes on Tick Data

High quality tick data has been sourced from Tick Data LLC at the cost of approximately 1000 USD. The focus for our research will be on S&P 500 E-mini futures, for the period 10 September 1997 - 13 February 2019. The S&P 500 E-Mini futures data is the set which de Prado regularly references in his work and by using the same set we create a natural way to benchmark our implementations.

A link to Tick Data LLC is provided: TickDataLLC

Trade Data

File Layout

Tick Data delivers its futures trade data in compressed, comma-delimited ASCII text files.

Prior to **Jul-1-2003**: Tick Data's historical futures data contains only day-session pit trading activity for all markets that were not electronic-only (i.e. had pit-only or pit and electronic trading). Electronic-only markets have partial night session trading (i.e. e-Mini trading days begin at 12:00am and close on the day session close), but do not have trading volume. Prior to **Jun-30-2003**, some of our data is time stamped to the second (HH:MM:SS), but most is time stamped to the minute (HH:MM). While additional fields can be outputted via the included TickWrite software, the as-traded data contains five (5) fields:

1. Date
2. Time
3. Price (always filtered price)
4. Volume (always zero)
5. Market Flag ('P' or 'E' for Pit or Electronic trades.

738 lines (737 sloc) | 146 KB

Raw | Blame | History

Create Financial Data Structures: mlfinlab

The purpose of this notebook is to act as a tutorial to bridge the gap between idea and implementation. In particular we will be looking at how to create the various financial data structures and how to format your data so that you can make use of the mlfinlab package.

For this tutorial we made use of the sample data provided by TickWrite LLC. Using S&P500 E-mini futures. https://s3-us-west-2.amazonaws.com/tick-data-s3/downloads/ES_Sample.zip

Note: we have also included the zip file in this repo as ES_Trades.csv.zip. You will need to unzip this file to use it in this notebook.

In [1]:

```
import mlfinlab as ml

import numpy as np
import pandas as pd
from scipy import stats
import seaborn as sns
from statsmodels.graphics.tsaplots import plot_acf
import matplotlib.pyplot as plt

%matplotlib inline
```

In [2]:

```
# Read data
data = pd.read_csv('./tutorial_data/ES_Trades/ES_Trades.csv')
data.head()
```

Out[2]:

	Symbol	Date	Time	Price	Volume	Market Flag	Sales Condition	Exclude Record Flag	Unfiltered Price
0	ESU13	09/01/2013	17:00:00.083	1640.25	8	E	0	NaN	1640.25
1	ESU13	09/01/2013	17:00:00.083	1640.25	1	E	0	NaN	1640.25
2	ESU13	09/01/2013	17:00:00.083	1640.25	2	E	0	NaN	1640.25
3	ESU13	09/01/2013	17:00:00.083	1640.25	1	E	0	NaN	1640.25
4	ESU13	09/01/2013	17:00:00.083	1640.25	1	E	0	NaN	1640.25

Event Based Sampling

Filter on Events

- Structural Breaks
- Microstructural information
- Corporate actions
- Dates: Weekend bias

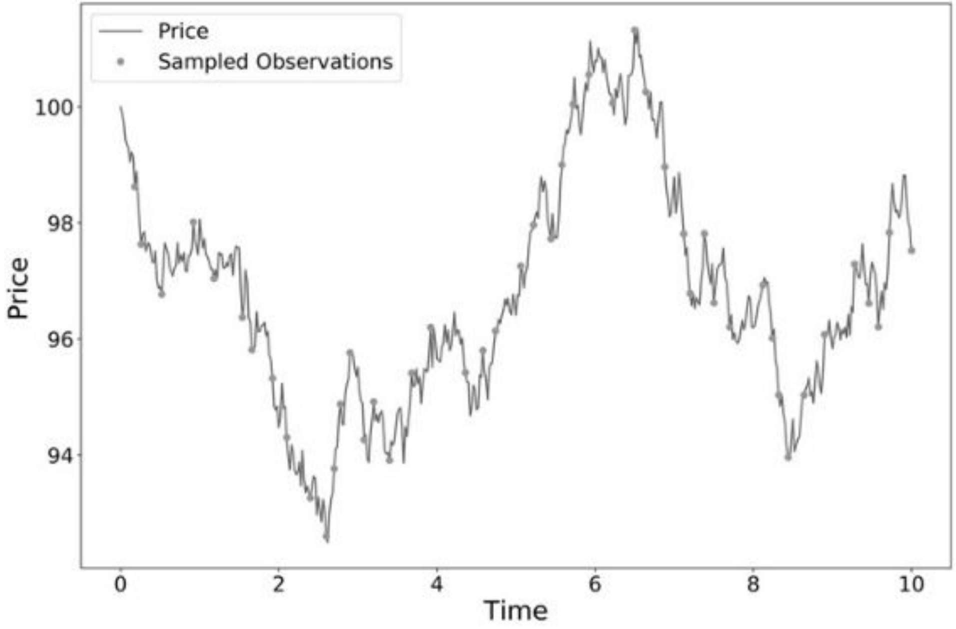
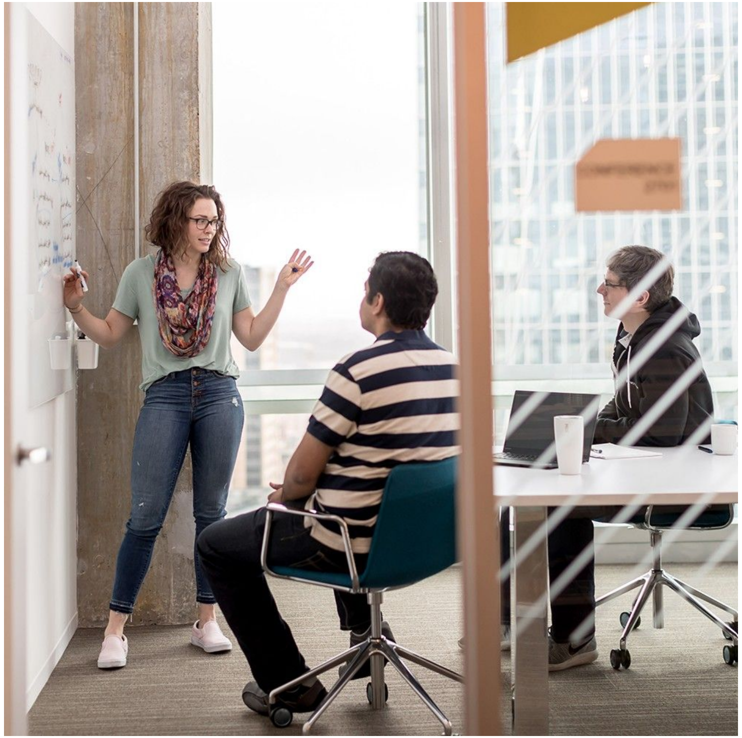


FIGURE 2.3 CUSUM sampling of a price series

Paradigm Shift



1. Focus on process: statistical properties, sampling, feature engineering, feature importance, ensembling.
2. Avoid research through backtesting.
3. Many-to-one models vs many-to-many
4. Trading approach vs investing
5. Keeping track of the number of trials run (Deflated Sharpe ratio)
6. Models can contain features from:
   a. Price action (microstructure)
   b. Fundamental (Accounting ratios)
   c. Alternative data (Satelites)
   d. Dimensionality reduction

Problem: Multiproduct series using a volume clock.

1. Sample using only one assets volume
2. Create a new metric like: (v1 + v2) / 2

GridGain

HUDSON AND THAMES

Additional Material

413 lines (412 sloc) | 36.9 KB

Raw | Blame | History

- By: Oleksandr Proskurin
- Email: proskurinolexandr@gmail.com
- Reference: Advances in Financial Machine Learning, Marcos Lopez De Prado, pg 40

ETF trick use case. SPX/EuroStoxx hedging implementation

Let's see how mlfinlab etf trick is used to solve the exercise 2.3 from Chapter 2. For this exercise we use daily SPY and EUROSTOXX futures data and EUR/USD exchange rates from https://www.investing.com/. Hedging weights are recalculated on a daily basis

In [3]:
```
#imports
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import datetime as dt
from mlfinlab.multi_product.etf_trick import ETFTrick
```

In [4]:
```
def generate_cov_mat(row):
    """
    Forms covariance matrix from current data frame row using 'rolling_cov',
    'rolling_spx_var' and 'rolling_eur_var' column values
    """
    cov = row['rolling_cov']
    spx_var = row['rolling_spx_var']
    euro_var = row['rolling_euro_var']
    return np.matrix([[spx_var, cov], [cov, euro_var]])
```

In [5]:
```
# Snippet 2.1 from a book
def pca_weights(cov, riskDist=None, risk_target = 1.):
    """
    Calculates hedging weights using covariance matrix(cov), risk distribution(riskDist) and ris
k_target
    """
    eVal, eVec = np.linalg.eigh(cov)
    indices = eVal.argsort()[::-1]
    eVal, eVec = eVal[indices], eVec[:, indices]
    if riskDist is None:
        riskDist = np.zeros(cov.shape[0])
        riskDist[-1] = 1.
    loads = risk_target * (riskDist/eVal)**.5
    wghts = np.dot(eVec, np.reshape(loads, (-1,1)))
    return wghts
```

Data preprocessing

371 lines (370 sloc) | 179 KB

Raw | Blame | History

- By: Proskurin Oleksandr
- Email: proskurinolexandr@gmail.com
- Reference: Advances in Financial Machine Learning, Marcos Lopez De Prado, pg 30, https://towardsdatascience.com/financial-machine-learning-part-0-bars-745897d4e4ba

In [10]:
```
from IPython.display import Image
```

Imbalance bars generation algorithm

Let's discuss imbalance bars generation on example of volume imbalance bars. As it is described in Advances in Financial Machine Learning book:

First let's define what is the tick rule:

For any given t, where p_t is the price associated with t and v_t is volume, the tick rule b_t is defined as:

$$b_t = \begin{cases} b_{t-1}, & \Delta p_t \\ |\Delta p_t| / \Delta p_t, & \Delta p_t \neq \end{cases}$$

Tick rule is used as a proxy of trade direction, however, some data providers already provide customers with tick direction, in this case we don't need to calculate tick rule, just use the provided tick direction instead.

Cumulative volume imbalance from 1 to T is defined as:

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

T is the time when the bar is sampled.

Next we need to define $E_0[T]$ as expected number of ticks, the book suggests to use EWMA of expected number of ticks from previously generated bars. Let's introduce the first hyperparameter for imbalance bars generation: **num_prev_bars** which corresponds to window used for EWMA calculation.

Here we face the problem of first bar generation, because we don't know expected number of ticks with no bars generated. To solve this we introduce the second hyperparameter: **expected_num_ticks_init** which corresponds to initial guess for expected number of ticks before the first imbalance bar is generated.

Bar is sampled when:

$$|\theta_t| >= E_0[T] * [2v^+ - E_0[v_t]]$$

To estimate $2v^+ - E_0[v_t]$ (expected imbalance) we simply calculate EWMA of volume imbalance from previous bars, that is why we need to store volume imbalances in *imbalance array*, the window for estimation is either **expected_num_ticks_init** before the first bar is sampled, or expected number of ticks($E_0[T]$) * **num_prev_bars** when the first bar is generated.

Note that when we have at least one imbalance bar generated we update $2v^+ - E_0[v_t]$ only when the next bar is sampled not on every trade observed

Algorithm logic

Thank You