A PIPELINE FOR BENCHMARKING FORECASTS IN FINANCIAL TIME SERIES DATA (*PnL Markouts*)

Mihai Cucuringu

February 6, 2026

Contents

1 Introduction

Having a common benchmarking pipeline is useful for

- putting everyone on the same page in terms of evaluation of predictions made
- comparing results across different studies, in terms of the same metrics
- avoid having to reinvent the wheel every time one is faced with the task of evaluating predictions made against various future target returns, potentially in a (often intraday) time-dependent manner (this will be made clear in the following sections).

2 Loading data - daily prices and volumes

- Basic data is available at the level of daily prices and volumes during 2000 - 2020, and can be loaded from the shared location in the format */YYYY/YYYYMMDD.csv* and also */YYYY/YYYYMMDD.data* (for ease of loading in R, the variable with the data frame is called A).
- For example, loading this in R amounts to

  load(file = '/2018/20181228.data');

  and doing a head and tail on the data frame A will display the following information.

```
    ticker   open   high    low  close    volume AdjCLoseY div split prevOpen prevClose prevVolume prevAdjCLoseY prevAdjClose
SPY    SPY 249.56 250.19 247.47 249.92 144299400 247.61964   0     1   249.58    247.75  153100200      245.46964       247.75
IWM    IWM 133.72 134.05 131.80 133.90  29173400 133.07487   0     1   132.48    132.86   35994400      132.04131       132.86
EEM    EEM  39.50  39.54  38.93  39.06  72972200  38.76444   0     1    39.19     39.24   72869100       38.94308        39.24
TLT    TLT 120.65 121.56 120.46 121.51  17409000 119.97957   0     1   120.40    121.05    9879100      119.52536       121.05
USO    USO   9.63   9.71   9.44   9.66  28417400   9.66000   0     1     9.54      9.53   22803400        9.53000         9.53
GLD    GLD 120.98 121.26 120.83 121.25   8449400 121.25000   0     1   120.80    121.06    6864700      121.06000       121.06
     ticker   open   high    low  close  volume AdjCLoseY div split prevOpen prevClose prevVolume prevAdjCLoseY prevAdjClose
ZBRA   ZBRA 157.07 159.33 155.98 159.23  409100 159.23000   0     1   156.98    155.97     344800      155.97000       155.97
ZEUS   ZEUS  14.41  14.58  13.94  14.27   72000  14.23299   0     1    14.13     14.44      76400       14.40255        14.44
ZION   ZION  40.49  40.96  39.94  40.74 2575600  40.23342   0     1    40.67     40.43    2558600       39.92728        40.43
ZIXI   ZIXI   5.65   5.83   5.65   5.73  247800   5.73000   0     1     5.39      5.60     403200        5.60000         5.60
ZTS     ZTS  85.27  85.59  84.60  85.54 1485200  85.10727   0     1    84.83     84.49    1797300       84.06258        84.49
ZUMZ   ZUMZ  19.09  19.24  18.69  19.17  254200  19.17000   0     1    19.00     19.01     245500       19.01000        19.01
[1] 1634   14
```

Figure 1: First and last rows in the daily matrix of price and volume information, for 2018-12-28.

- The data frame has 1634 instruments, one row for each separate instrument. The ticker is a unique identifier for each instrument. The columns include the *open*, *high*, *low*, *close* prices on the day of, as well as traded *volume*, and *split* and *div* (dividend) information (which you can safely ignore for now). The *close* price in there is already adjusted for dividends (though there is a separate column in there, denoted *AdjCLoseY* which should roughly match the *close* column; *AdjCLoseY* is data which Yahoo directly disseminates as being adjusted for splits and dividends). The data frame has additional columns for previous day's *open*, *close*, *volume*, and *AdjCLoseY*, for ease of manipulation. You can safely ignore the last column *prevAdjClose* which should be identical to the *prevClose* column.
- These instruments can be extracted from the daily files of prices, alternatively you can load a single file with the entire relevant historical data:
  - /Matrix_Format_Over_Time/SP1500__Intersect/CLOSE_20000128_20190701.csv for the daily close prices
  - /Matrix_Format_Over_Time/SP1500__Intersect/VolumeDollar_20000128_20190701.csv for the daily dollar volumes

  The files are also available in $.data$ R format.
- Upon loading the CLOSE variable from the above location, we display below, for the first 6 instruments, their respective close prices for the first 3 days and the last 3 days, available in history:
- Note that this file only contains the **intersection** of the instruments available in the universe on each data; which explains the decrease from the over 1600 instruments available for a specific day in Figure 1 to only 1029 in the current figure. Note that there are 4881 days available in history, hence the number of columns.
- on a given day, when you load up past history (and future returns), it might be good to have a function that performs a sanity check on the daily returns of each individual instrument, and drop a name if there is a too large of a fraction of entries that is either equal to zero or are extremely large (eg, too many days with over

```
> CLOSE[ 1:6, c(1:3, 4879 :4881)  ]
     20000128  20000131  20000201 20190627 20190628 20190701
SPY 135.87500 139.56250 140.93750   291.50   293.00   295.66
XLF  18.22705  18.73477  18.83631    27.21    27.60    27.93
XLB  23.46875  23.32812  23.75000    58.01    58.50    58.95
XLK  48.93750  50.56250  51.62500    78.05    78.04    79.28
XLV  30.25000  30.10930  30.50000    92.33    92.64    93.10
XLI  27.00000  27.18750  27.00000    76.61    77.42    77.49
> dim(CLOSE)
[1] 1029 4881
```

Figure 2: A subset of the full matrix of daily close prices.

100% returns). For example, it might be a good idea to have a function which takes as input a matrix RETS (of size n by d, indexing n stocks across d days), and do a pre-processing step, which also depends on the horizon over which the returns have been computed. For example, if working with daily returns, one could drop the days for which more than n/10 stocks have a zero returns in that respective column; and drop the stocks (rows) for which more than d/2 of its days have a zero returns; and further drop the stocks (rows) for which more than d/10 of its days have a return larger than a certain pre-specified threshold (eg, 100% return). This all typically hints at unreliable data, and the specific thresholds should be chosen realistic (for example, if working with minutely returns, it may be OK to have a large number of zero returns, but not so OK if working with daily returns. On the other hand, if working with weekly returns, it may be OK to have a large number of returns be over 100%, but not OK if working with minutely returns).

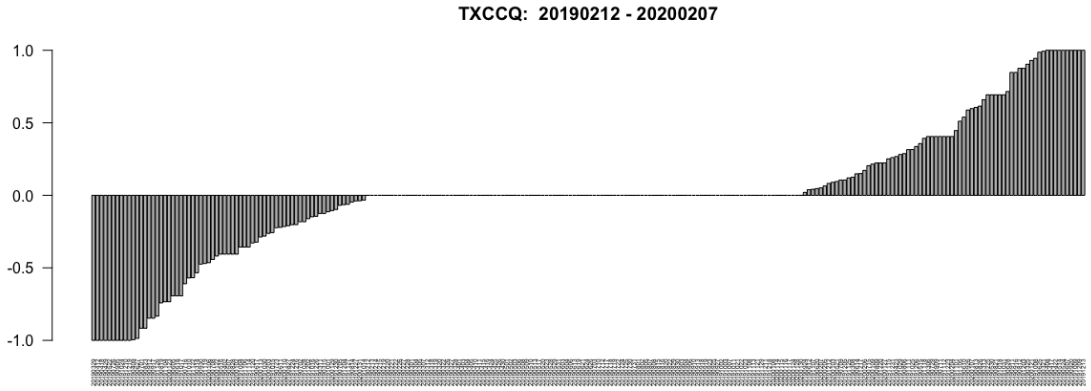


Figure 3: Example with suspicious data.

2.1 Minutely data

In many studies, it is useful to capture data at a more granular level, and customary to snap data on a minutely basis, as it could be used to infer data for higher granularity (move from 1-minute studies to 15-minute, or 30-minute or hourly).

At every time t (minute of the day, starting with 9:31 until 16:00), the data below captures future returns from time t onward, at various future time horizons $h = \{1, 5, 10, 30, 60, 90, 180\}$ minutes.

$fret\_t\_close$ denotes the future return from time t until the official close price of the day.

In addition, the data also contains the future overnight CLOP (close-to-open) and CLCL (close-to-close) returns, though for a given instrument ticker, these two columns are constant, since these returns do not depend on t (as illustrated in Figure 5).

Finally, the future returns come in two flavors: raw returns (RR) and market-excess returns (MR), with the latter return being discussed in a later section.

It is also worth pointing out that the **past** h-minutely returns can also be inferred from this data set, by appropriately shifting the time series.

Link to data repository. Minutely data is available here

https://www.dropbox.com/sh/rsf5c6u84gy506e/AADKcMF3vYnT5XXvazWi1Hiba?dl=0

for about 380 NASDAQ-listed instruments, for every day during 2018-2019 (about 2 x 250=500 days, with one file per day). Note that each file may not unzip if you attempt to do so, but could be loaded directly in R or Python.

	Time	Ticker	fret_1_RR	fret_5_RR	fret_10_RR	fret_30_RR	fret_60_RR	fret_90_RR	fret_180_RR	fret_tClose_RR	fret_CLOP_RR	fret_CLCL_RR
1	09:31	AAP	0.0023	0.0120	0.0155	0.0327	0.0469	0.0553	0.0594	0.0463	0.0031	0.0090
2	09:31	AAPL	-0.0014	-0.0019	0.0003	0.0058	0.0091	0.0093	0.0084	0.0136	0.0016	-0.0002
3	09:31	ABBV	0.0003	0.0034	0.0037	0.0042	0.0083	0.0154	0.0093	0.0118	0.0014	0.0155
4	09:31	ABC	0.0022	0.0016	0.0038	0.0092	0.0079	0.0083	0.0084	0.0130	-0.0068	0.0037
5	09:31	ABT	0.0005	0.0033	0.0057	0.0129	0.0102	0.0100	0.0108	0.0095	0.0034	0.0022
6	09:31	ACN	-0.0020	-0.0013	0.0018	0.0017	0.0010	0.0024	-0.0008	0.0028	-0.0055	0.0046

Figure 4: Example of future returns for several instruments at 9:31:00 on 2018-01-02.

	Time	Ticker	fret_1_RR	fret_5_RR	fret_10_RR	fret_30_RR
2	09:31	AAPL	-0.0014	-0.0019	0.0003	0.0058
386	09:32	AAPL	-0.0008	0.0003	0.0017	0.0080
770	09:33	AAPL	-0.0009	0.0019	0.0036	0.0090
1154	09:34	AAPL	0.0014	0.0042	0.0051	0.0098
1538	09:35	AAPL	-0.0002	0.0024	0.0041	0.0087
1922	09:36	AAPL	0.0007	0.0022	0.0051	0.0089
2306	09:37	AAPL	0.0009	0.0014	0.0046	0.0084
2690	09:38	AAPL	0.0013	0.0016	0.0040	0.0078
3074	09:39	AAPL	-0.0004	0.0009	0.0033	0.0071
3458	09:40	AAPL	-0.0004	0.0017	0.0039	0.0074

Figure 5: Example of future returns for AAPL for the first 10 minutes of the day, on 2018-01-02.

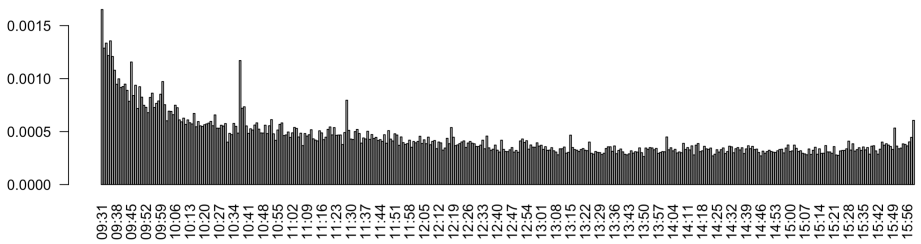


Figure 6: Barplot of the average absolute value returns, of about 380 instruments on the above day.

3 Getting started with a small data set

- In terms of getting started with a smaller data set, from the shared data, one can subset from the top of the daily prices files the following 9 ETFs {XLF, XLB, XLK, XLV, XLI, XLU, XLY, XLP, XLE} that correspond to the industry sectors of the US market. If you want to slightly enlarge the set, you could add the very top instruments {SPY, IWM, EEM, TLT, USO, GLD} (they capture some more "macro" variables, such as SP500 index, oil, gold, interest rates).
- you can either extract this subset from the daily prices files, or you can access a single file with the entire relevant historical data:
  - `/Matrix_Format_Over_Time/ETFs_and_MACRO__Union/CLOSE_20000128_20190701.csv` for the daily close prices
  - `/Matrix_Format_Over_Time/ETFs_and_MACRO__Union/VolumeDollar_20000128_20190701.csv` for the daily dollar volumes

  The files are also available in $.data$ R format.

4 Miscellaneous

1. In general, it is customary to work with log-returns, which can be defined as follows. The return of instrument (i.e. stock) i, between times t_1 and t_2 is given by

$$\text{pret}_i^{(t_1,t_2)} = \log \frac{P_{i,t_2}}{P_{i,t_1}} \tag{1}$$

   where $P_{i,t}$ denotes the price of instrument i at time t. Computing the returns in vectorized form, for a set of instruments, can be done in one line of code. For example, assuming that the matrix CLOSE of size $n \times T$ contains the daily closing prices of n stocks for T days, we can compute the previous 1-day past returns for all days (starting with the second day) and all stocks with the following line of R code

PRETS_1day = log(CLOSE[, 2: T)] / CLOSE[, 1: (T-1)]);

2. "quantile analysis" - quantile portfolios, meant to validate/show how much additional value is there in the magnitude of the prediction/signal. This involves mapping the individual entries in the signal vector (of size $n \times 1$) to their respective (signed) quantile rank values. Normalization by volatility could also be taken into account.
3. If you are thinking about tracking temporal changes in your embeddings/low-dimensional representations, then procrutes analysis

`https://www.mathworks.com/help/stats/procrustes.html`

   may be useful, as it was in this paper for example

`https://www.turing.ac.uk/research/research-projects/discovering-polarity-change-word-meanings`.

4. Always beware of outliers! A good practice when dealing with financial data is to winsorize your (past) returns (for example, at 1% for both tails).
   - When preprocessing stock returns, the choice between winsorizing across time or across stocks depends on the modeling objective. Winsorizing across time (ie, capping extreme returns for each stock over its own time series) is ideal for time-series forecasting or volatility modeling. This approach prevents idiosyncratic outliers from distorting the dynamics of individual stocks, particularly in models that rely on the statistical properties of a single asset's return distribution. It is well-suited for per-stock models like ARIMA, GARCH, or rolling regressions, where stability and robustness over time are more important criteria.
   - Alternatively, winsorizing across stocks at each point in time is better suited for cross-sectional models, such as factor-based strategies, ranking algorithms, or cross-sectional regressions. This method reduces the impact of extreme stock-specific shocks that may otherwise dominate signal construction or distort factor loadings. It improves the reliability of rankings, z-scores, and alpha signals at each date.
   - In practice, both approaches can be applied at different preprocessing stages depending on whether the goal is to stabilize time-series behavior or to construct clean cross-sectional comparisons.

Table 1: Choice of winsorization approach by modeling objective

Goal / Context	Better Winsorization Strategy
Time-series modeling per stock	Across time (per stock)
Cross-sectional alpha signals	Across stocks (per time)
Portfolio sorting, factor models	Across stocks (per time)
Robustifying volatility models	Across time (per stock)
Combined use case	Consider both, depending on preprocessing step

5 Performance metrics

5.1 Future returns

One is typically interested in evaluating forecasts (often denoted as **alphas** in the literature) made **only** using prior historical data, against future returns (often denoted as **targets**). For ease of notation and referencing, we use the abbreviation **fret** to denote forward looking returns, and include in the name of the return additional information to reflect the type of target being used.

The future returns/targets can be simply raw returns or various decompositions of the returns, and we shall see such examples further below. We can let $\text{fret}_{i,t}^{(h)}$, or for simplicity of notation, $\text{f}_{i,t}^{(h)}$ denote the raw return of instrument i at time t with future horizon h by

$$\text{f}_{i,t}^{(h)} = \log \frac{P_{i,t+h}}{P_{i,t}} \tag{2}$$

where $P_{i,t}$ denotes the price of instrument i at time t. For example h can denote a 3-day future horizon, or a 5-second or 30-minute intraday future horizon, depending on the scenario considered. Similarly, t can index either time in terms of dates (eg, t denotes 2016-07-22) or intraday time (eg, $t = 13:30:30$).

5.2 Beta to the Market and Market Excess Returns

One often uses the SP500 as a proxy for the entire market return. According to Wikipedia *The S&P 500, or just the S&P, is a stock market index that measures the stock performance of 500 large companies listed on stock exchanges in the United States. It is one of the most commonly followed equity indices, and many consider it to be one of the best representations of the U.S. stock market.*. The SP500 index has a corresponding ETF (Exchange Traded Fund) which essentially trades just like any other regular stock. The symbol of this instrument is denoted by SPY, and let us consider its return as

$$\text{f}_{SPY,t}^{(h)} = \log \frac{P_{SPY,t+h}}{P_{SPY,t}} \tag{3}$$

With this in mind, for any instrument/stock i, one can then consider its future **market excess return**

$$\tilde{\text{f}}_{i,t}^{(h)} = \text{f}_{i,t}^{(h)} - \beta \text{f}_{SPY,t}^{(h)} \tag{4}$$

For simplicity, one often assumes $\beta = 1$ across all instruments, though there are various techniques to infer the betas from historical data.

5.3 Why should we hedge and benchmark against the Market Excess Returns?

In general, aiming to evaluate "alphas" given by either

- predictions per-se (as a result or fitted historical models), or
- simply signed variables (obtained as outputs from various unsupervised learning techniques, for example)

against future raw return f, may not necessarily be the best way going forward. The reason for this is that the raw return contains a built-in component, construed as the *market return*, which, if left in there and not accounted for, makes one's life harder and often gives one the false impression that the end results of one's study are not satisfactory or weak.

A simple example Let us consider a simple toy example. Suppose one has available a piece of alpha given by market news sentiment, just to take one example, and this alpha is strongly positive hinting that the price of stock X might strongly increase over the next day. Suppose that you go ahead and buy the stock, but for whatever reasons, the overall

market goes down by 3% over the next day, due to a number of external factors (political events, coronavirus, war, etc). Because of this, and because the overall price of the market also has an influence on the stock X, let us assume that P_X the stock price of X goes down by 2%. If you evaluate your alpha α_X against f_X you would lose money (your alpha was positive, but the price of X went down). However, if you evaluate your alpha against the market excess return $\tilde{f}_X$, then you would win money, because the market excess return

$$\tilde{f}_X = f_X - f_{SPY} = -2\% - (-3\%) = +1\% \tag{5}$$

is also positive, just like the sign of your alpha. Note this execution style is *tradeable*, as in practice it would correspond to buying \$100 worth of X, and (short) selling \$100 worth of SPY. Your long position in X would lose you \$2, while your short position in SPY would win you \$3, thus overall you win \$1, which corresponds indeed to the 1% win from the market excess return in (5), applied to the \$100 position. Here we have seen a simple toy example of *hedging*; note that the disadvantage is that hedging does not come for free, as, in addition to the \$100 needed to long X, we also needed another \$100 to (short)-sell SPY, though in practice there are various cheap way to perform such a hedge (but let us not worry about this aspect there, as it is not the main point of the present document).

Finally, note that such hedging can also

- limit your gains $\alpha_X = +1$, and $f_X = +3\%$, but $f_{SPY} = 2\%$, thus $\tilde{f}_X = f_X - f_{SPY} = 3\% - 2\% = 1\%$ (thus you only gain 1%, not the 3% you would have gained in the unhedged position)
- and sometimes go against you. For example, if $\alpha_X = +1$, and $f_X = +2\%$, but $f_{SPY} = 5\%$ then $\tilde{f}_X = f_X - f_{SPY} = 2\% - 5\% = -3\%$, meaning that you end up losing money, even if your bet was on the right side of the move of X.

The main takeaway message here is that hedging leads to higher Sharpe Ratios, which is the topic of the next section.

5.4 Sharpe Ratio

For the evaluation of the performance, you could use some standard metrics which you could implement in a few lines of code. If s is your variable or *"alpha"* of interest (i.e., a forecast/prediction/quantity of interest, eg, perhaps the above ϵ vector), then the "PnL" (Profit and Loss) of your variable on a given day t is simply given by

$$PnL_t = \sum_{i=1}^{n} \text{sign}(s_i^{(t)}) * \text{fret}_i^{(t)}, \quad t = 1, \dots, T, \tag{6}$$

where $s_i^{(t)}$ denotes your forecast on day t for instrument i. The sum is across all the instrument, and $\text{fret}_i^{(t)}$ is some future return of instrument i on day t (you could, for example, explore different forward looking horizon windows such as $h \in \{1, 3, 5, 10\}$). For instance $\text{fret}_i^{(t)} = \log \frac{P_{i,t+h}}{P_{i,t}}$, and you could add a superscript to the PnL calculation to indicate its dependency on h, i.e. $PnL_t^{(h=1)}$ for 1-day out PnL.

Once the PnL time series has been computed for all available days in the study, (in a rolling window approach) and thus PnL is a time series of length T, one computes the corresponding (annualized) Sharpe Ratio (SR) as

$$SR = \frac{\text{mean}(PnL)}{\text{stdev}(PnL)} * \sqrt{252}, \tag{7}$$

where the scaling is due to the fact that there are 252 trading days within a year.

5.4.1 Sharpe Ratio Statistical Significance Testing

If daily returns were normally distributed, the corresponding Sharpe Ratio would resemble a t-statistic. Because the underlyings' returns are not normally distributed (they are known to have heavy tails and negative skewness), [1] proposes a modified statistic which is given by

$$\frac{\text{SR}(v)}{\sqrt{\frac{1-\text{skewness}(v)\text{SR}(v)+(\text{kurtosis}(v)-1)(\text{SR}(v)^2/4)}{T-1}}},$$

Note that this test regards the Sharpe ratio and not the annualized Sharpe ratio. SR denote the Sharpe ratio, v is the vector of returns, and T is the length of v. The code to do a simple two tailed test would look something like that

```
library(e1071)
sharpe_test <- function(v)
{
    sharpe_ratio = mean(v) / sd(v)
    Ti = length(v)
    g3 <- skewness(v);
    g4 <- kurtosis(v)
    p1 <- pnorm(sharpe_ratio / sqrt((1 - g3*sharpe_ratio + (g4-1)*
(sharpe_ratio^2/4))/(Ti-1)))
    return(min(p1,1-p1)*2)
}
```

If you want a one-tailed test you would modify the p-value respectively / also depending if you are looking for a significantly profitable / losing strategy.

The second test regards the comparison of two trading strategies, similar to a two sample t-test. For this, we can use [2] who uses a bootstrapping method. This one is implemented in the PeerPerformance package (sharpeTesting function).

Main references to consider here are:

1 Bailey D, Lopez de Prado M. The Deflated Sharpe Ratio: Correcting for Selection Bias, Backtest Overfitting, and Non-Normality. The Journal of Portfolio Management. 2014 09;40:94-107

2 Ledoit O, Wolf M. Robust performance hypothesis testing with the Sharpe ratio. Journal of Empirical Finance. 2008;15(5):850-9.

5.5 Scaling & betsize due to market cap

$$PnL_t = \sum_{i=1}^{n} \text{sign}(s_i^{(t)}) \cdot \text{fret}_i^{(t)} \cdot b_i^{(t)}, \quad t = 1, \dots, T, \tag{8}$$

where $b_i^{(t)}$ denotes the bet-size associated to instrument i on day t. Typically, $b_i^{(t)}$ is set proportional to the (recent historical) liquidity of the respective instrument i at time t, while not exceeding a certain notional (eg USD) amount, in order to prevent a few very liquid stocks to dominate the entire portfolio. For example, the bet-size could be taken as the minimum of ϕ (eg $\phi = 0.5\%$) percent of the median-21-day notional daily volume, and $A = \$200,000$. In other other

$$b_i^{(t)} = \min\{0.5\% \times \text{median daily USD traded volume}, \$200,000\}. \tag{9}$$

Yet another alternative approach is to (also) weight each position (i.e. betsize) by the magnitude of the forecasts $|s_i^{(t)}|$, which essentially amounts to

$$PnL_t = \sum_{i=1}^{n} s_i^{(t)} \cdot \text{fret}_i^{(t)} \cdot b_i^{(t)}, \quad t = 1, \dots, T, \tag{10}$$

This essentially can be thought as (8) except that $b_i^{(t)}$ is replaced by $\tilde{b}_i^{(t)} = b_i^{(t)} \cdot |s_i^{(t)}|$. Warning: be careful for outliers in your $s_i^{(t)}$ estimates.

5.6 (Global) Average PnL per unit traded (PPD, PPT)

In the financial literature, a typical performance measure is the average return per unit traded, in percentage. For example, one is typically interested in the annualized return of the portfolio. For instance, if at time t_0 the available capital is \$100, and the cumulative PnL at time t_{250} (thus after a full year, with about 250 business days), the cumulative PnL is \$10, then the annualized return was 10%. Recall that 1% amounts to 100 basis points (100 bpts); easier to write 1 bpts rather than 1% or 0.0001 when displaying results. Roughly speaking, an annualized return of 10% translates to about 4 bpts per day (since $4 \times 250 = 1000$ bpts, which amounts to 10%).

It is often the case that we are interested in the average PnL per day in basis points, because it can be easier placed into the context of fees and transaction costs. For example, the average spread (between the best bid price and the best ask price) throughout the day, is about 4 bpts, for the most liquid equity instruments in the US markets. Also, it is often the case that exchanges (NYSE, NASDAQ, LSE, etc) charge a fee for each dollar traded on their platform (eg, 0.3 bpts),

which they earn on every single transaction made (yes, it is extremely profile to be an exchange, they make money no matter what – it's a very high Sharpe Ratio business ☺).

The **total portfolio size** on a given day is defined as

$$B_t = \sum_{i=1}^{n} b_i^{(t)}, \tag{11}$$

where $b_i^{(t)}$ denotes the betsize of instrument i on day t.

PPD = PnL per Dollar traded (when we are assuming that betsizes are different for each instrument)

$$PPD = \frac{\sum_{t=1}^{T} PnL_t}{\sum_{t=1}^{T} B_t}. \tag{12}$$

Essentially, the PPD is telling us how much would you earn for each 1 dollar traded in the markets (excluding transaction costs). In more simplistic studies, one may choose to ignore sizing effects, which leads us to the following simplified notion of PPT.

PPT = PnL per Trade (when we are assuming that all the betsizes are equal to \$1, i.e., $b_i^{(t)} = 1, i = 1, \ldots, n$ across all days; thus $B_t = n, \forall t = 1, \ldots, T$)

$$PPT = \frac{\sum_{t=1}^{T} PnL_t}{Tn} \tag{13}$$

5.7 (Local) Average PnL and daily PPD/PPD

Alternatively, we could also compute the daily PPD

$$DailyPPD_t = \frac{PnL_t}{B_t}. \tag{14}$$

and then compute the Mean or Median of this daily sequence.

Significant differences between the global PPD version and the local PPD version may arise when there are large variations in the numbers of instruments each day and/or the daily values of B_t.

Motivation

Note: the current pipeline for PnL analysis can also be used in a setting where the cross-sectionality of the data on a given day is replaced by multiple events of a single instrument. In other words, in the standard setup explore in the previous Section 5.6, in any given day t, we have a single event for each instrument - i.e., we have a single forecast $s_i^{(t)}$ for each instrument $i = 1, \ldots, n$, and we would like to compute the PnL against future returns $fret_i^t$. The task here was to the perform an analysis of the column-vector of forecasts $s^{(t)}$ against the column-vector of future realized returns $fret^t$, each of size n.

Now imagine a setting where the universe consists only of a single instrument, but there are many forecasts $s_i^{(t)}$ made within day t, and we would like to analysis the pnl of this set of forecasts against certain future realized returns. For example, we could be making a prediction every hour (thus i would index hours of the day, eg. in the crypto market which never, there would be $n = 24$ such observations within a day), and we would like to assess their performance against future 15-minute realized returns.

Finally, consider the scenario when the index i indexes certain events within a day, for example, suppose we make a prediction every time a trade occurs or every time a specific condition on the LOB is met); in this setting, $s_i^{(t)}$ denotes the i^{th} forecast (trade) made on day t, and the goal is to predict the corresponding future 1 minute realized return $fret_i^t$. In such a setting, the number of events within a day can greatly vary across days (eg, there could be days with 10,000 trade events, but also days with 100,000 trade events). In this scenario, it is also important to consider both the global and the local version of the PPD quantities (and similarly for PPT).

5.8 Cumsum PnL

Can consider (either or both)

- cumsum of the daily PnL (in notional dollar terms). See for example Figure 7 (d)
- cumsum of the daily return PnL as a percentage (thus in basis points). This pertains to the scenario when you have a varying number of instruments each day, or a varying notional size (weights) on each day.

5.9 Decile Portfolios

One is often interested in understanding the performance of the forecasts, as a function of their magnitudes. To this end, one typically considers only a subset (eg, top k% strongest in magnitude forecasts) of the universe of instruments, usually referred to as *decile* portfolios in the literature [3]. Assume for example a universe of 500 stocks, in the SP500 index. A quantile-based analysis simply constructs and evaluates portfolios composed of stocks which fall in a specific quantile bucket, or above a quantile threshold. The plots in Figure zzz consider a scenario with 6 different types of forecasts (denoted as ABC-1,2,3 and XYZ-1,2,3, which for example, may correspond to using two methods ABC and XYZ, each with 3 different parameter combinations). The colors denote the quantile portfolios, meaning they denote the fraction of the stocks traded, based on their magnitude

- red bars corresponding to 100% denote the entire portfolio of 500 stocks (denote this as the qrank_1 portfolio)
- the green bars corresponding to 75% mean that we only consider the top 75% largest in magnitude forecasts (thus a portfolio of size 375) (denote this as the qrank_2 portfolio)
- blue bars denote the top 50% (thus a portfolio of size 250) (denote this as the qrank_3 portfolio)
- black bars denote the top 25% (thus a portfolio of size 125) (denote this as the qrank_4 portfolio)

Plot (a) in Figure 7 denotes the Sharpe ratio, across all methods, for each quantile portfolio. Similarly, Plot (b) shoes the PPT (PnL per Trade), i.e. for each stock traded, assuming a bet size of \$1 uniformly across all stocks. Plot (c) denotes the annualized PnL, obtained by simply multiplying the results in Plot (b) by 250 trading days in a year. For example, the 25% quantile portfolio of method ABC-1 achieves about 3.5 bpts. Annualized, this amounts to $3.5 \times 252 = 882$ bpts, which translates to about 8.82% annualized return.

Finally, plot (d) in the same Figure 7 shows the cumulative return across the full history, across all methods, for the full 100% portfolio.

5.10 Additional statistics to capture

For each quantile portfolio, we can compute additional daily performance statistics, in addition to those previously mentioned.

In the previous section, we have computed the following performance statistics for each quantile portfolio

- PnL := total PnL for the instruments in each quantile portfolio
- sizeNotional := total number of dollars allocated for each quantile portfolio
- PPD := "Pnl Per Dollar" (:= PnL / notionalAmount)
- nrInstr := total number of instruments (for which $s_i \neq 0$) in each quantile portfolio. A few comments are in place:
  - This number is typically constant across days, and is the corresponding fraction of instruments. This number would typically take values in the set $\{500, 375, 250, 125\}$ depending on the quantile portfolio.
  - however, because often data is faulty/NA or populated with zeroes (ie many of the values s_i are zeroes), we only want to count the number of nonzero and finite $s_i \neq 0$. Due to this fact, nrInstr could typically take values $\{470, 335, 220, 100\}$
  - for distributions of there the values s_i take only a small set of distinct values, or many of the values s_i are equal, it not uncommon for the nrInstr could typically take values $\{470, 450, 430, 3900\}$ (to see this, imagine a setting where $s_i \in [-1, 1]$ but most of the signals s_i actually take value ± 1)

In addition to these daily summary statistics, we could also capture the following summary statistics across each quantile portfolio of instruments (here, we again ignore instruments for which $s_i = 0$):

- corr_SP := the Spearman correlation between the signals s_i and the target fret_i
- dcor := the Distance Correlation between the signals s_i and the target fret_i (there is also the signed distance correlation version)
- hitRatio := fraction of instruments i for which sign(s_i) = sign(fret_i)
- longRatio := the fraction of instruments for which sign(s_i)=1 (this will essentially tell us if a signal s is biased towards taking positive values more often than negative values)
- R2 := the Rsquare from the regression of fret against s

- t_stat := t-statistics estimators from the regression of fret against s

If you imagine capturing your daily summary into a 4D array, called

$$.STATS := 4D\ array \tag{15}$$

its dimensions could look as follows

1. (TypeStatistic): "pnl", "ppd", "sizeNotional", "nrInstr", "corr_SP", "hitRatio", "longRatio"
2. (TypeSignal): "fcst_Algo_A", "fcst_Algo_B", "news_Sent_C"
3. (TypeQrank): "qr_1", "qr_2", "qr_3", "qr_4"
4. (TypeTarget): "fret_CLOP_RR", "fret_CLOP_MR", "fret_CLCL_RR", "fret_CLCL_MR"

In other words, for each combination of fret and forecast (indexed by dimension 2, respectively 4), we compute the a number of summary performance statistics (such as "pnl", "ppd", "sizeNotional", "nrInstr", etc) for each quantile portfolio. We repeat this procedure for each day, and finally compute the aggregated performance statistics such as those shown in Figure 7: (a) Annualized Sharpe Ratio; (b) PPT (average (across days) Pnl per trade); (c) annualized average return.

Similar plots to those in Figure 7 could be shown for the average (across days) of the remaining performance statistics: "sizeNotional", "nrInstr", "corr_SP", "hitRatio", "longRatio", etc. In addition, one could also plot the distribution (across days) of these statistics (one distribution per quantile), instead of just capturing their average values.

Finally, one could then plot cumulative PnL plots across time, such as those depicted in Figure 7 (d), which plot (for a given fixed target fret) the 6 different signals/methods for the quantile rank 1 slide of the data.

But one could slice the data in various ways and plots the following. For example, possible combinations include

- fix the quantile rank (eg, qr_3) and the target (eg, fret_CLCL), and plot cumPnl (across time) of the different methods (eg, "fcst_Algo_A", "fcst_Algo_B", "news_Sent_C"). As in Figure 7 (d)
  - this is useful to compare the performance of different types of signals/methodologies/algorithms/etc for a given qrank x target combination.
- fix the signal method (eg,"news_Sent_C") and the target (eg, fret_CLCL), and plot cumPnl (across time) of the different quantile ranks (eg, "qr_1", "qr_2", "qr_3", "qr_4")
  - this is useful to compare the performance of different quantiles of a given signal x target combination, and allow s to establish where the larger magnitude signals are indeed stronger.
- fix the signal method (eg,"news_Sent_C") and the qrank (eg, "qr_2"), and plot cumPnl (across time) of the different targets (eg, "fret_CLOP_RR", "fret_CLOP_MR", "fret_CLCL_RR", "fret_CLCL_MR")
  - this latter representation could be useful to understand the difference in performance of a given signal at various types of targets (future returns), for example RR (raw-return) versus MR (market-excess-return); or evaluate the performance/decay in forecasting power as we increase the forecasting horizon, since we sometimes may have the targets as follows "fret_CLCL_MR_1d", "fret_CLCL_MR_3d", "fret_CLCL_MR_5d", "fret_CLCL_MR_10d" (at 1,3,5,10 days in the future)

6 Data inputs for your evaluation pipeline

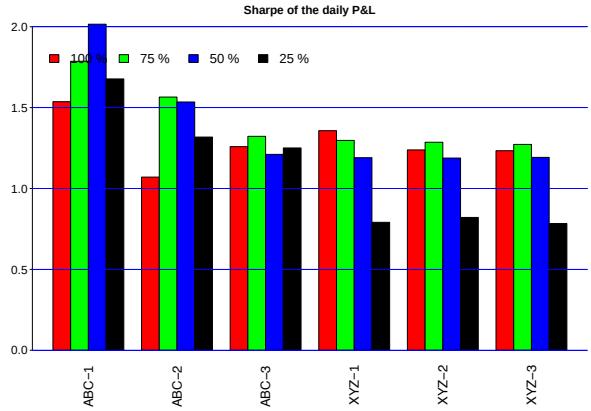
Ideally, your pipeline should be able to take as input a set of paths to where daily files reside, and compute all the above statistics. For example, you should have one file for each day (it's good practice to name these files as 20200722.csv for example, or 20200722.csv.gz), with the following columns:

- $ticker$ (ie., name of the instrument; unique identifier)
- $\{signal\_1, signal\_2, \ldots, signal\_k\}$ for the k different signals to compare (they could all have different names, though it is a good idea to have all the signals start with the prefix "signal_" or "fcst_" (or your favorite prefix) to facilitate for later grep'ing of all the signal columns via a certain substring)
- $\{fret\_1, fret\_2, \ldots, fret\_m\}$ for the m different targets (again, they can have different names fret_RR_CLOP, fret_RR_CLCL, etc - it's just good practice to have them all start by fret to facilitate later grep'ing by a certain substring)

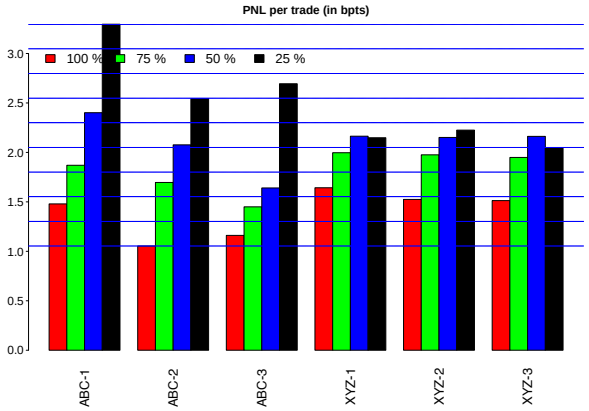
You could organize your files as follows:

- have a first folder called RAW_DATA/ which contains all the daily above files YYYYMMDD.csv
- have a second folder called DAILY_SUMMARIES/ which contains all the files with the daily summary performance statistics as captured in the 4d array (15).
  - Note this STATS 4d array could be saved to a pickle python file, or a .data R file.
  - The STATS 4d array could also, in principle, be saved into a csv file, but some pre-processing/pivoting would have to be done to transform the 4d array into a 2d representation: one way of doing this could be to have 5 columns TypeStatistic, TypeSignal, TypeQrank, TypeTarget, value and record each entry in the 4d tensor as one row in a 2d matrix.

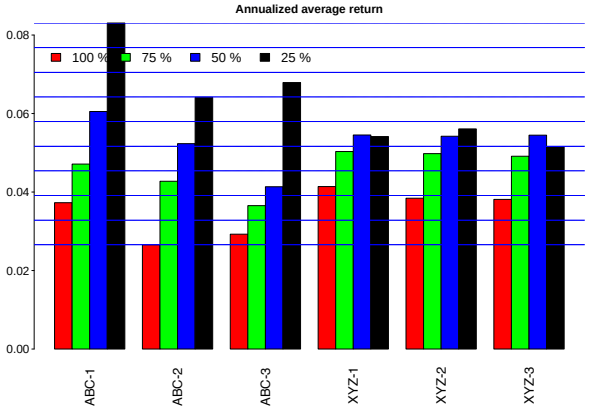
It is good practice to have your data ready in place in the above format, before attempting to implement the summary performance statistics pipeline detailed in the previous Section 5



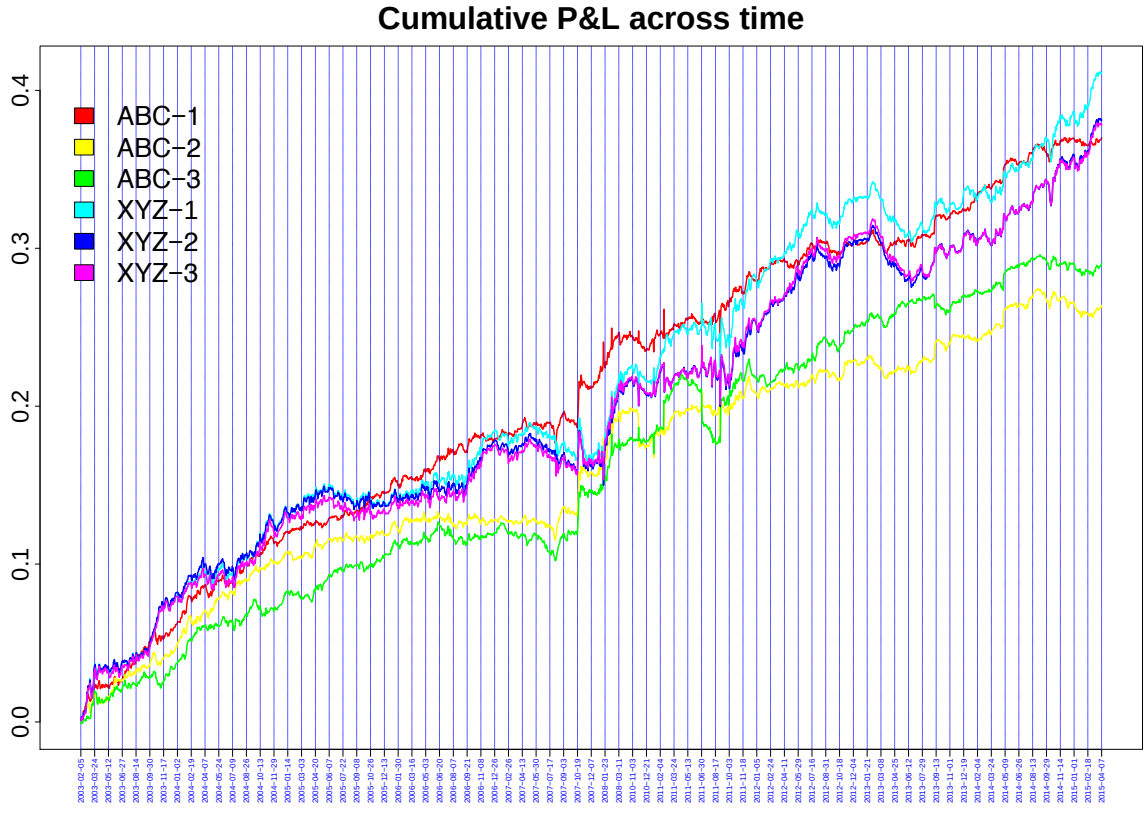
(a) Sharpe Ratio across all methods, for all four quantile portfolios.



(b) PPT (Pnl Per Trade).



(c) Annualized average return.



(d) Cumulative PnL plot across time, across all 6 methods considered, for the entire portfolio (corresponding to the red bars in the above barplot)).

Figure 7: Example of summary statistics relevant for financial data.

7 Coding practices

Writing clean reusable code:

https://mathdatasimplified.com/2023/05/11/python-clean-code-6-best-practices-to-make-your-python-functi

References

[1] David Bailey and Marcos Lopez de Prado. The deflated sharpe ratio: Correcting for selection bias, backtest overfitting, and non-normality. *The Journal of Portfolio Management*, 40:94–107, 09 2014.

[2] Oliver Ledoit and Michael Wolf. Robust performance hypothesis testing with the sharpe ratio. *Journal of Empirical Finance*, 15(5):850–859, 2008.

[3] Eugene F Fama and Kenneth R French. The Cross-Section of Expected Stock Returns. *Journal of Finance*, 47(2):427–65, June 1992.