Financial Data Structures

Source: Marco López de Prado, Advances in Financial Machine Learning, Wiley, 2018

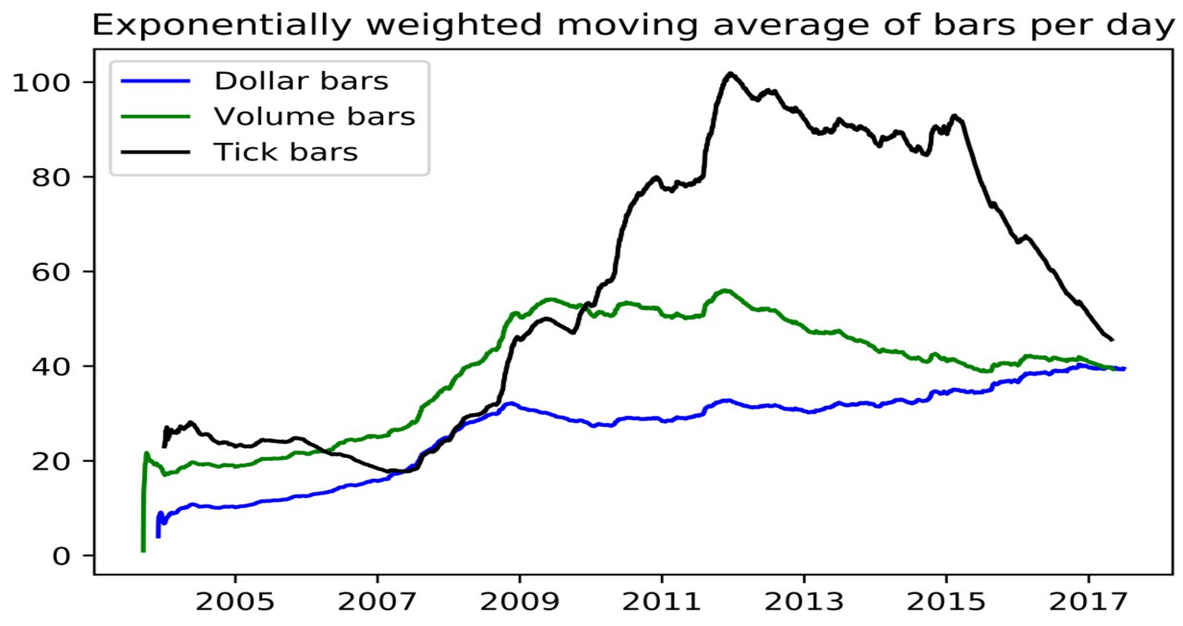
Essential Types of Financial Data

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

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 runs bars.
    - Order imbalance bars.
    - Entropy bars.

Example 1: 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.

Example 2: Dollar Imbalance Bars (0/2)

Consider a sequence of ticks $\{(p_t, v_t)\}_{t=1,\dots,T}$, 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 $\{b_t\}_{t=1,\dots,T}$ 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.

Example 2: 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

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

- 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.

Example 2: 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.

Labeling

Source: Marco López de Prado, Advances in Financial Machine Learning, Wiley, 2018

Labeling in Finance

- Virtually all ML papers in finance label observations using the fixed-time horizon method.
- Consider a set of features $\{X_i\}_{i=1,\dots,I}$, drawn from some bars with index $t = 1,\dots,T$, where $I \leq T$. An observation X_i is assigned a label $y_i \in \{-1,0,1\}$,

$$y_i = \begin{cases} -1 & \text{if } r_{t_{i,0},t_{i,0}+h} < -\tau \\ 0 & \text{if } \left| r_{t_{i,0},t_{i,0}+h} \right| \leq \tau \\ 1 & \text{if } r_{t_{i,0},t_{i,0}+h} > \tau \end{cases}$$

where τ is a pre-defined constant threshold, $t_{i,0}$ is the index of the bar immediately after X_i takes place, $t_{i,0} + h$ is the index of h bars after $t_{i,0}$, and $r_{t_{i,0},t_{i,0}+h}$ is the price return over a bar horizon h.

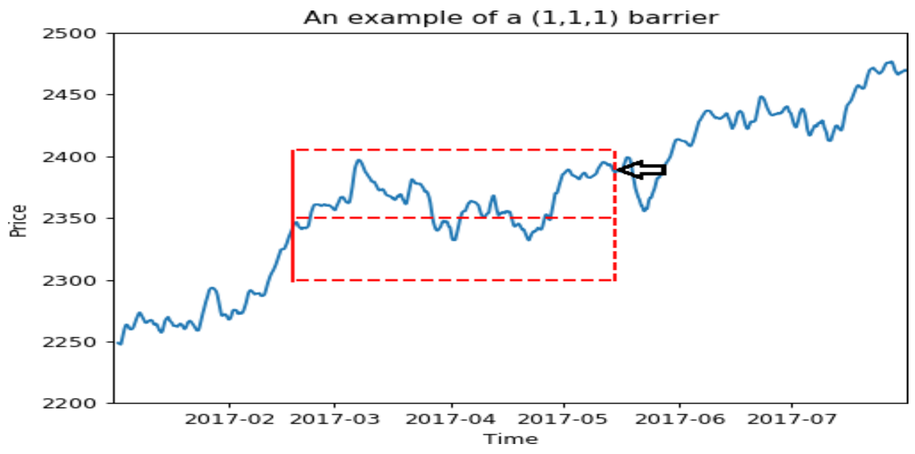
- Because the literature almost always works with time bars, h implies a fixed-time horizon.

Caveats of the Fixed Horizon Method

- There are several reasons to avoid such labeling approach:
  - Time bars do not exhibit good statistical properties.
  - The same threshold τ is applied regardless of the observed volatility.
    - Suppose that $\tau = 1E - 2$, where sometimes we label an observation as $y_i = 1$ subject to a realized bar volatility of $\sigma_{t_{i,0}} = 1E - 4$ (e.g., during the night session), and sometimes $\sigma_{t_{i,0}} = 1E - 2$ (e.g., around the open). The large majority of labels will be 0, even if return $r_{t_{i,0},t_{i,0}+h}$ was predictable and statistically significant.
- A couple of better alternatives would be:
  - Label per a varying threshold $\sigma_{t_{i,0}}$, estimated using a rolling exponentially-weighted standard deviation of returns.
  - Use volume or dollar bars, as their volatilities are much closer to constant (homoscedasticity).
- But even these two improvements miss a key flaw of the fixed-time horizon method: The *path* followed by prices. We will address this with the Triple Barrier Method.

The Triple Barrier Method

- It is simply unrealistic to build a strategy that profits from positions that would have been stopped-out by the fund, exchange (margin call) or investor.
- The Triple Barrier Method labels an observation according to the first barrier touched out of three barriers.
    - Two horizontal barriers are defined by profit-taking and stop-loss limits, which are a dynamic function of estimated volatility (whether realized or implied).
    - A third, vertical barrier, is defined in terms of number of bars elapsed since the position was taken (an expiration limit).
- The barrier that is touched first by the *price path* determines the label:
    - Upper horizontal barrier: Label 1.
    - Lower horizontal barrier: Label -1.
    - Vertical barrier: Label 0.



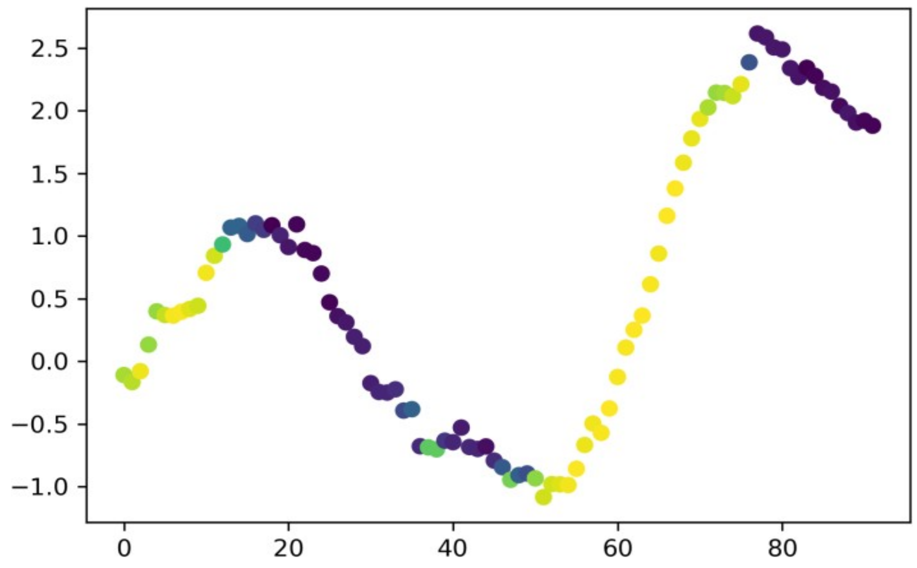
Trend Scanning Method

Consider a series of observations $\{x_t\}_{t=1,\dots,T}$, where x_t may represent the price of a security we aim to predict. We wish to assign a label $y_t \in \{-1,0,1\}$ to every observation in x_t, based on whether x_t is part of a downtrend, no-trend, or an uptrend. One possibility is to compute the t-value (t_{β_1}) associated with the estimated regressor coefficient (β_1) in a linear time-trend model,

$$x_{t+l} = \beta_0 + \beta_1 l + \varepsilon_{t+l}$$

$$t_{\beta_1} = {}^{\beta_1}/_{\sigma_{\beta_1}}$$

where σ_{β_1} is the standard deviation of β_1, and $l = 0, \dots, L-1$, and L sets the look-forward period, with $L \leq t$. Different values of L lead to different t-values. To solve this indetermination, we can try a set of values for L, and pick the value that maximizes $|t_{\beta_1}|$. In this way, we assign to x_t the most significant trend observed in the past, out of multiple possible look-forward periods.

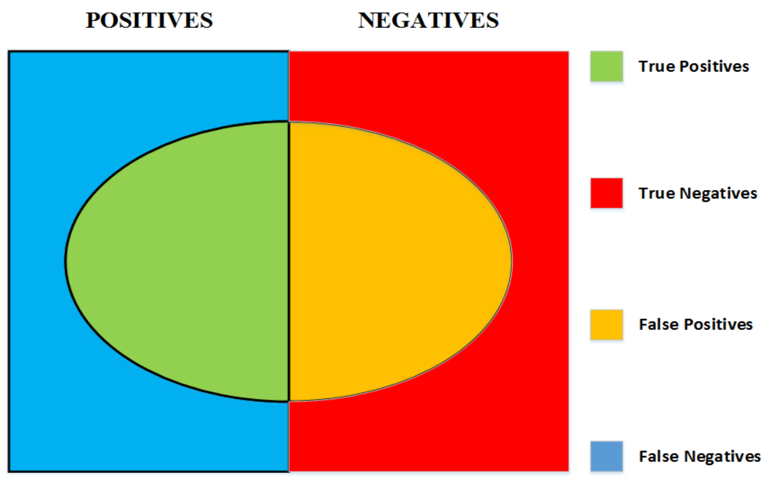


Meta-Labeling

Source: Marco López de Prado, Advances in Financial Machine Learning, Wiley, 2018

Turning a Weak Predictor into a Strong Predictor

- Suppose that you have a model for making a buy-or-sell decision:
  - You just need to learn the ***size*** of that bet, which includes the possibility of no bet at all (zero size).
  - This is a situation that practitioners face regularly. We often know whether we want to buy or sell a product, and the only remaining question is how much money we should risk in such bet.
  - Meta-labeling: Label the outcomes of the primary model as 1 (gain) or 0 (loss). See Sections 3.6-3.8 of AFML.
  - The goal is not to predict the market. Instead, the goal is to predict the success of the primary model.



- The primary model forecasts the direction of the trade.
- Meta-labeling builds a secondary ML model that learns how to use a primary exogenous model: evaluate if the agent should trade or not based on the primary model, and what is the size of the trade.
- Meta-labeling is particularly useful when outcomes are asymmetric. In those cases, giving up some recall (TP/(TP+FN)) in exchange for improving the precision (TP/(TP+FP)) (concentrating on good forecasts when market is up) can yield a significant improvement in the Sharpe ratio.

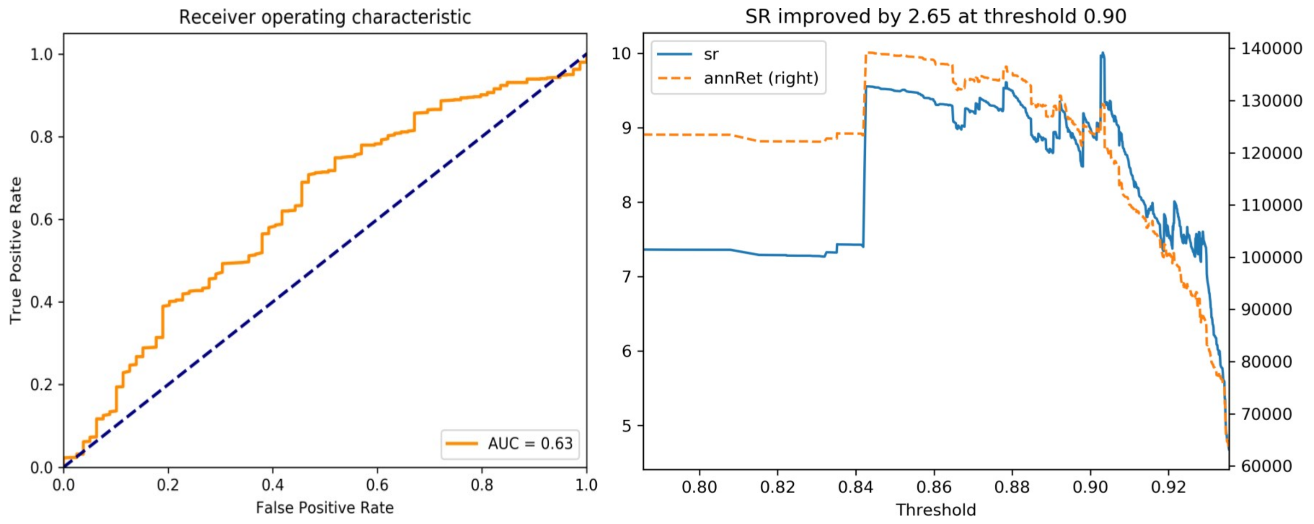
Why Meta-Labeling Works

- The Sharpe ratio associated with a binary outcome can be derived as

$$\theta[p,n,\pi_-,\pi_+] = \frac{(\pi_+ - \pi_-)p + \pi_-}{(\pi_+ - \pi_-)\sqrt{p(1-p)}}\sqrt{n}$$

where $\{\pi_-, \pi_+\}$ determine the payoff from negative and positive outcomes, p is the probability of a positive outcome, and n is the number of outcomes per year (see Section 15.3 of AFML).

- When $\pi_+ \gg -\pi_-$, it may be possible to increase $\theta[.]$ by increasing p at the expense of n.



The primary model determines $\{\pi_-, \pi_+\}$, and the secondary model regulates $\{p, n\}$.

In this example, a strategy's Sharpe ratio increased by 2.65 thanks to Meta-Labeling's ability to avoid the largest losses.

How to use Meta-labeling

- Meta-labeling is particularly helpful when you want to achieve higher F1-scores:
  - First, we build a model that achieves high recall (TP/(TP+FN)), even if the precision (TP/(TP+FP) is not particularly high.
  - Second, we correct for the low precision by applying meta-labeling to the positives identified by the primary model.
- Meta-labeling is a very powerful tool in your arsenal, for three additional reasons:
  - ML algorithms are often criticized as *black boxes*. Meta-labeling allows you to build a ML system on a white box.
  - The effects of *overfitting* are limited when you apply meta-labeling, because ML will not decide the side of your bet, only the size.
  - Achieving high accuracy on small bets and low accuracy in large bets will ruin you. As important as identifying good opportunities is to *size bets* properly, so it makes sense to develop a ML algorithm solely focused on getting that critical decision (sizing) right.
- Meta-labeling should become an essential ML technique for every discretionary hedge fund
  - It allows the seamless combination of discretionary inputs (primary model) with a quantitative overlay (secondary model).

Sample Weights

Source: Marco López de Prado, Advances in Financial Machine Learning, Wiley, 2018

Weighting observations by uniqueness (1/2)

- Two labels y_i and y_j are concurrent at *t* when both are a function of at least one common return, $r_{t-1,t} = \frac{p_t}{p_{t-1}} - 1$.

1. For each observation $t = 1, \dots, T$ we form a binary array, $\{1_{t,i}\}_{i=1,\dots,I}$, with $1_{t,i} \in \{0,1\}$, which indicates whether its outcome spans over return $r_{t-1,t}$.
2. We compute the number of labels concurrent at *t*, $c_t = \sum_{i=1}^{I} 1_{t,i}$.
3. The uniqueness of a label *i* at time *t* is $u_{t,i} = 1_{t,i} c_t^{-1}$.
4. The average uniqueness of label *i* is the average $u_{t,i}$ over the label's lifespan, $\bar{u}_i = \left(\sum_{t=1}^{T} u_{t,i}\right)\left(\sum_{t=1}^{T} 1_{t,i}\right)^{-1}$.

Weighting observations by uniqueness (2/2)

5. Sample weights can be defined as the sum of the attributed absolute log returns, $|r_{t_{i-1},t_i}|$, over the event's lifespan, $[t_{i,0}, t_{i,1}]$

$$\tilde{w}_i = \left| \sum_{t=t_{i,0}}^{t_{i,1}} \frac{r_{t-1,t}}{c_t} \right|$$

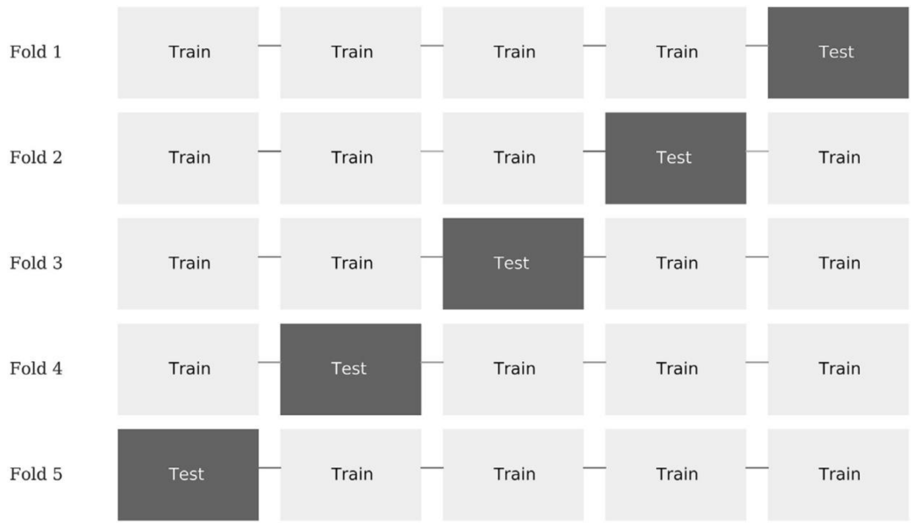
$$w_i = \tilde{w}_i I \left(\sum_{j=1}^{I} \tilde{w}_j \right)^{-1}$$

- The rationale for this method is that we weight an observation as a function of the absolute log returns that can be attributed *uniquely* to it.
- We can use these weights for sequential bootstrap (section 4.5).

Cross-Validation in Finance

Source: Marco López de Prado, Advances in Financial Machine Learning, Wiley, 2018

Why K-Fold Cross-Validation Fails in Finance



Leakage takes place when the training set contains information that also appears in the testing set. Consider a serially correlated feature X that is associated with labels Y that are formed on overlapping data:

- Because of the serial correlation, $X_t \approx X_{t+1}$.
- Because labels are derived from overlapping datapoints, $Y_t \approx Y_{t+1}$.

Therefore, by placing *t* and *t + 1* in different sets, information is leaked: When a classifier is first trained on (X_t, Y_t), and then it is asked to predict $\mathrm{E}[Y_{t+1} | X_{t+1}]$ based on an observed X_{t+1}, this classifier is more likely to achieve $Y_{t+1} = \mathrm{E}[Y_{t+1} | X_{t+1}]$ even if X is an irrelevant feature.

Note that, for leakage to take place, it must occur that $(X_{train}, Y_{train}) \approx (X_{test}, Y_{test})$, and it does not suffice that $X_{train} \approx X_{test}$ or even $Y_{train} \approx Y_{test}$.

Purged K-Fold CV

- One way to reduce leakage is to purge from the training set all observations whose labels overlap in time with those labels included in the testing set (*purging*).

- Consider a label Y_j that is a function of observations in the closed range $t \in [t_{j,0}, t_{j,1}]$, $Y_j = f\left[\left[t_{j,0}, t_{j,1}\right]\right]$.
    - For example, in the context of the triple barrier labeling method, it means that the label is the sign of the return spanning between price bars with indices $t_{j,0}$ and $t_{j,i}$, that is $\text{sign}\left[r_{t_{j,0}, t_{j,1}}\right]$.

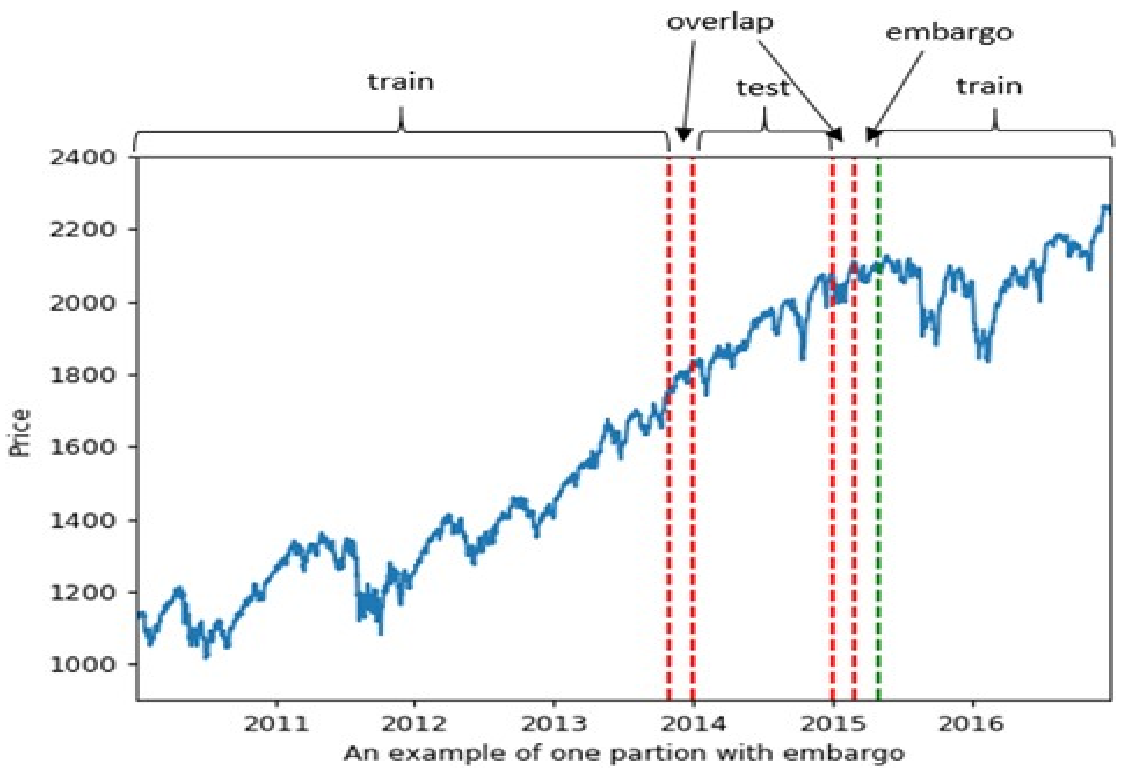
- A label $Y_i = f\left[\left[t_{j,0}, t_{j,1}\right]\right]$ overlaps with Y_j if any of the three sufficient conditions is met:

$$t_{j,0} \leq t_{i,0} \leq t_{j,1};\ t_{j,0} \leq t_{i,1} \leq t_{j,1};\ t_{i,0} \leq t_{j,0} \leq t_{j,1} \leq t_{i,1}$$

Embargoed K-Fold CV

- Since financial features often include series that exhibit serial correlation (like ARMA processes), we should eliminate from the training set observations that immediately follow an observation in the testing set (*embargo)*.
  - The embargo does not need to affect training observations prior to a test, because training labels $Y_i = f\left[\left[t_{i,0}, t_{i,1}\right]\right]$, where $t_{i,1} < t_{j,0}$ (training ends before testing begins), contain information that was available at the testing time $t_{j,0}$.
  - We are only concerned with training labels $Y_i = f\left[\left[t_{i,0}, t_{i,1}\right]\right]$ that take place immediately after the test, $t_{j,1} \leq t_{i,0} \leq t_{j,1} + h$.
- We can implement this embargo period h by setting $Y_j = f\left[\left[t_{j,0}, t_{j,1} + h\right]\right]$ before purging. A small value $h \approx .01T$, where T is the number of bars, often suffices to prevent all leakage.

Example: Purging and Embargoing



This plot shows one partition of the K-Fold CV.The test set is surrounded by two train sets, generating two overlaps that must be purged to prevent leakage.

Tofurther prevent leakage, the train observations immediately after the testing set are also embargoed.