The 10 Reasons Most Machine Learning Funds Fail

Marcos López de Prado

JPM 2018, 44 (6) 120-133
doi: https://doi.org/10.3905/jpm.2018.44.6.120
http://jpm.iijournals.com/content/44/6/120

Institutional Investor Journals
1120 Avenue of the Americas, 6th floor,
New York, NY 10036, Phone: +1 212-224-3589

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

EXHIBIT 1
The 10 Reasons Most Machine Learning Funds Fail

#	Category	Pitfall	Solution
1	Epistemological	The Sisyphus paradigm	The meta-strategy paradigm
2	Epistemological	Research through backtesting	Feature importance analysis
3	Data processing	Chronological sampling	The volume clock
4	Data processing	Integer differentiation	Fractional differentiation
5	Classification	Fixed-time horizon labeling	The triple-barrier method
6	Classification	Learning side and size simultaneously	Meta-labeling
7	Classification	Weighting of non-IID samples	Uniqueness weighting; sequential bootstrapping
8	Evaluation	Cross-validation leakage	Purging and embargoing
9	Evaluation	Walk-forward (historical) backtesting	Combinatorial purged cross-validation
10	Evaluation	Backtest overfitting	Backtesting on synthetic data; the deflated Sharpe ratio

with patterns. This fact, combined with the low signal-to-noise ratio that characterizes finance, all but ensures that careless users will produce false discoveries at an ever-greater speed. The goal of this article is to expose some of the most common errors made by ML experts when they apply their techniques on financial datasets. The following sections summarize those pitfalls (listed in Exhibit 1) and propose solutions. The interested reader may find a more detailed explanation by López de Prado [2018].

PITFALL #1: THE SISYPHUS PARADIGM

Discretionary portfolio managers (PMs) make investment decisions that do not follow a particular theory or rationale (if they had one, they would be systematic PMs). They consume raw news and analyses, but mostly rely on their judgment or intuition. They may rationalize those decisions based on some story, but anyone can always make a story up to justify any decision. Because no one fully understands the logic behind their bets, investment firms ask them to work independently from one another, in silos, to ensure diversification. If you have ever attended a meeting of discretionary PMs, you probably noticed how long and aimless those meetings can be. Each attendee seems obsessed about one particular piece of anecdotal information, and giant argumentative leaps are made without fact-based, empirical evidence. This does not mean that discretionary PMs cannot be successful. On the contrary, a few of them are. The point is, they cannot naturally work as a team. Bring 50 discretionary PMs together, and they will influence one another until eventually you are paying 50 salaries for the work of one. Thus, it makes sense for them to work in silos so they interact as little as possible.

Wherever I have seen that formula applied to quantitative or ML projects, it has led to disaster. The boardroom's mentality is, let us do with quants what has worked with discretionary PMs. Let us hire 50 PhDs and demand that each of them produce an investment strategy within six months. This approach tends to backfire because each PhD will frantically search for investment opportunities and eventually settle for either (1) a false positive that looks great in an overfit backtest or (2) standard factor investing, which may be an overcrowded strategy with a low Sharpe ratio but at least has academic support. Both outcomes will disappoint the investment board, and the project will be cancelled. Even if 5 of those PhDs make a true discovery, the profits will not suffice to cover the expenses of 50, so those 5 will relocate somewhere else, searching for a proper reward.

SOLUTION #1: THE META-STRATEGY PARADIGM

If you have been asked to develop ML strategies on your own, the odds are stacked against you. It takes almost as much effort to produce one true investment strategy as to produce a hundred, and the complexities are overwhelming: data curation and processing, high-performance computing infrastructure, software

development, feature analysis, execution simulators, backtesting, and the like. Even if the firm provides you with shared services in those areas, you are like a worker at a BMW factory who has been asked to build an entire car by using all the workshops around you. One week you need to be a master welder, another week an electrician, another week a mechanical engineer, another week a painter.... You will try, fail, and circle back to welding. How does that make sense?

Every successful quantitative firm of which I am aware applies the meta-strategy paradigm (López de Prado [2014]). Tasks of the assembly line are clearly divided into subtasks. Quality is independently measured and monitored for each subtask. The role of each quant is to specialize in a particular task, to become the best there is at it, while having a holistic view of the entire process. Teamwork yields discoveries at a predictable rate, with no reliance on lucky strikes. No particular individual is responsible for these discoveries because they are the outcome of team efforts, in which everyone contributes. Of course, setting up these financial laboratories takes time and requires people who know what they are doing and have done it before, but what do you think has a higher chance of success: this proven paradigm of organized collaboration or the Sisyphean alternative of having every single quant rolling an immense boulder up the mountain?

PITFALL #2: RESEARCH THROUGH BACKTESTING

One of the most pervasive mistakes in financial research is to take some data, run it through an ML algorithm, backtest the predictions, and repeat the sequence until a nice-looking backtest shows up. Academic journals are filled with such pseudo-discoveries, and even large hedge funds constantly fall into this trap. It does not matter if the backtest is a walk-forward (WF) out of sample. The fact that we are repeating a test over and over on the same data will likely lead to a false discovery. This methodological error is so notorious among statisticians that they consider it scientific fraud, and the American Statistical Association warns against it in its ethical guidelines (American Statistical Association [2016], Discussion #4). It typically takes about 20 such iterations to discover a (false) investment strategy subject to the standard significance level (false positive rate) of 5%.

SOLUTION #2: FEATURE IMPORTANCE ANALYSIS

Suppose that you are given a pair of matrices (X, y) that, respectively, contain features and labels for a particular financial instrument. We can fit a classifier on (X, y) and evaluate the generalization error through cross-validation. Suppose that we achieve good performance. The next natural step is to try to understand what features contributed to that performance. Maybe we could add some features that strengthen the signal responsible for the classifier's predictive power. Maybe we could eliminate some of the features that are only adding noise to the system. Most critically, understanding feature importance opens up the proverbial black box.

We can gain insight into the patterns identified by the classifier if we understand what source of information is indispensable to it. This is one of the reasons why the black box mantra is somewhat overplayed by the ML skeptics. Yes, the algorithm has learned without us directing the process (that is the whole point of ML!) in a black box, but that does not mean that we cannot (or should not) take a look at what the algorithm has found. Hunters do not blindly eat everything their smart dogs retrieve for them, do they? Once we have found what features are important, we humans can learn more by conducting a number of experiments. Are these features important all the time, or only in some specific environments? What triggers a change in importance over time? Can those regime changes be predicted? Are those important features also relevant to other, related financial instruments? Are they relevant to other asset classes? What are the most relevant features across all financial instruments? What is the subset of features with the highest rank correlation across the entire investment universe? This is a much better way of researching strategies than the foolish backtest cycle. Remember, feature importance is a research tool and backtesting is not.

PITFALL #3: CHRONOLOGICAL SAMPLING

To apply ML algorithms on 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*.

Although time bars are perhaps the most popular among practitioners and academics, they should be avoided for two reasons. First, markets do not receive information at a constant time interval. The hour following the open is much more active than the hours around noon (or the hours around midnight in the case of futures). As biological beings, it makes sense for humans to organize their day according to the sunlight cycle. However, today's markets are operated by algorithms that trade with loose human supervision, for which chronological intervals are not necessarily meaningful (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, such as serial correlation, heteroskedasticity, and non-normality of returns (Easley, López de Prado, and O'Hara [2012]). Generalized autoregressive conditional heteroskedasticity models were developed, in part, to deal with the heteroskedasticity associated with incorrect sampling.

SOLUTION #3: THE VOLUME CLOCK

We can avoid the two problems described earlier by forming bars as a subordinated process of trading activity. This approach is sometimes referred to as the *volume clock* (Easley, López de Prado, and O'Hara [2013]). For instance, *dollar bars* are formed by sampling an observation every time a predefined market value is exchanged. Of course, the reference to dollars is meant to apply to the currency in which the security is denominated, but no one refers to euro bars, pound bars, or yen bars.

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 to sample 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. Exhibit 2 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 affect the amount of ticks and volumes, such as issuing new shares or buying back existing shares (a very common practice since the Great Recession of 2008). Dollar bars tend to be robust to 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).

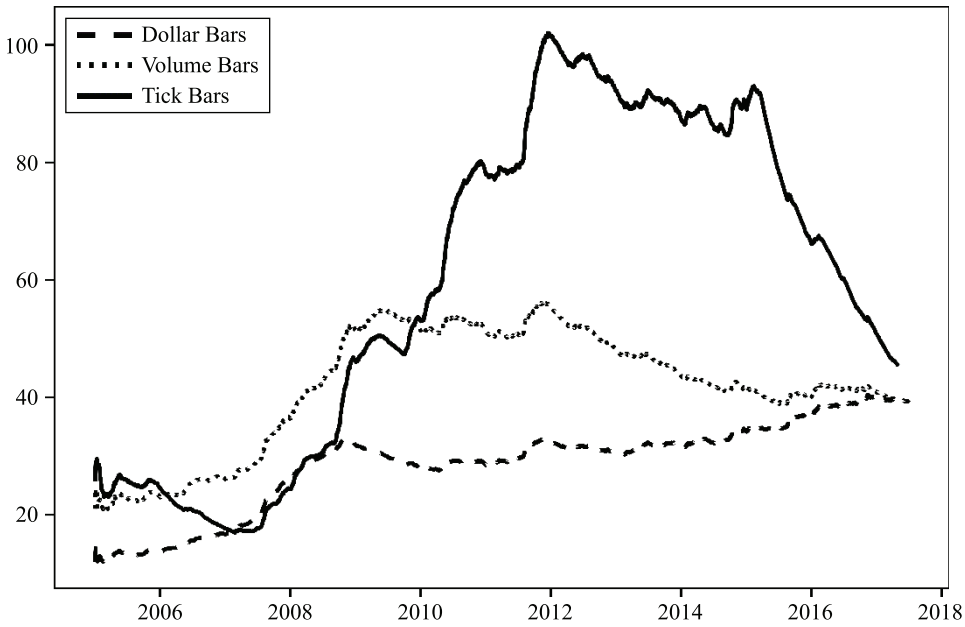
There are more sophisticated types of bars, which sample observations as a function of the arrival of asymmetric information, but they are beyond the scope of this article. See Chapter 2 in *Advances in Financial Machine Learning* by López de Prado [2018] for a discussion of these.

PITFALL #4: INTEGER DIFFERENTIATION

It is common in finance to find nonstationary time series. What makes these series nonstationary is the presence of memory (i.e., a long history of previous levels that shift the series' mean over time). To perform inferential analyses, researchers need to work with invariant processes, such as returns on prices (or changes in log-prices), changes in yield, or changes in volatility. These data transformations make the series stationary, at the expense of removing all memory from the original series (Alexander [2001, Chapter 11]). Although stationarity is a necessary property for inferential purposes, it is rarely the case in signal processing that we wish all memory to be erased, because that memory is the basis for the model's predictive power. For example, equilibrium (stationary) models need some memory to assess how far the price process has drifted from the long-term expected value in order to generate a forecast. The dilemma is that returns are stationary but memoryless,

EXHIBIT 2

Average Daily Frequency of Tick, Volume, and Dollar Bars



and prices have memory but are nonstationary. The question arises: What is the minimum amount of differentiation that makes a price series stationary while preserving as much memory as possible?

Supervised learning algorithms typically require stationary features because we need to map a previously unseen (unlabeled) observation to a collection of labeled examples and infer from them the label of that new observation. If the features are not stationary, we cannot map the new observation to a large number of known examples. However, stationarity does not ensure predictive power. Stationarity is a necessary, nonsufficient condition for the high performance of an ML algorithm. The problem is, there is a trade-off between stationarity and memory. We can always make a series more stationary through differentiation, but it will be at the cost of erasing some memory, which will defeat the forecasting purpose of the ML algorithm.

Returns are just one (and, in most cases, suboptimal) kind of price transformation among many other possibilities. Part of the importance of cointegration methods is their ability to model series with memory, but why would the particular case of zero differentiation deliver best outcomes? Zero differentiation is as arbitrary as one-step differentiation. There is a wide region between these two extremes (fully differentiated series on one hand and zero differentiated series on the other).

SOLUTION #4: FRACTIONAL DIFFERENTIATION

Virtually all the financial time series literature is based on the premise of making nonstationary series stationary through integer differentiation (see Hamilton [1994] for an example). But why would integer 1 differentiation (like the one used for computing returns on log-prices) be optimal?

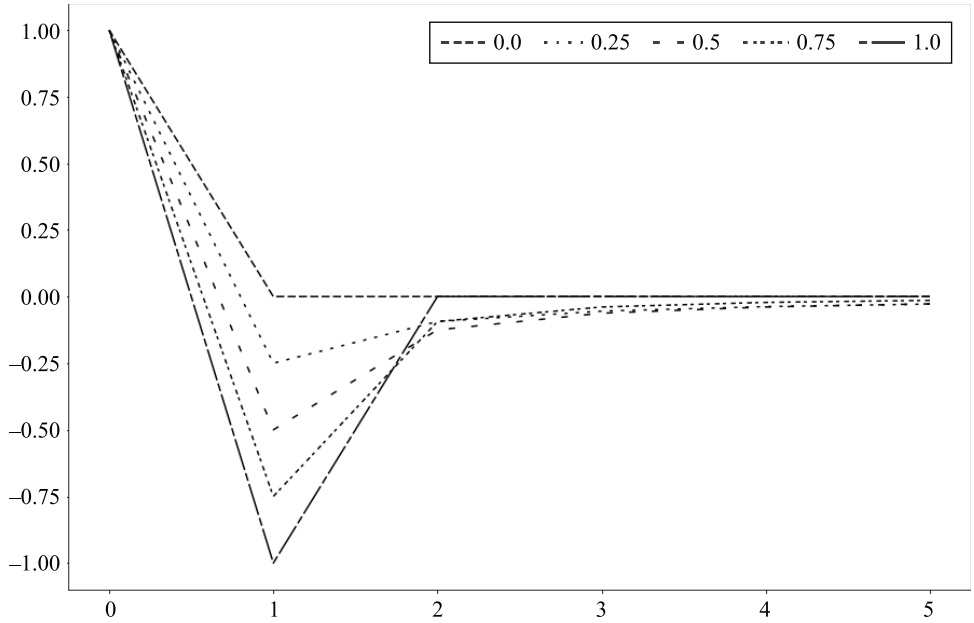
Fractional differentiation (FracDiff) allows us to generalize the notion of returns to noninteger (positive real) differences d. Given a time series of observations $\{x_t\}_{t=1,\ldots,T}$, the FracDiff transformation of order d at time t is $\tilde{x}_t = \sum_{k=0}^{\infty} \omega_k x_{t-k}$, with $\omega_0 = 1$ and

$$\omega_k = -\omega_{k-1} \frac{d-k+1}{k}$$

For the derivation and meaning of this equation, please see López de Prado [2018, Chapter 5]. For instance, when $d = 0$, all weights are 0 except for $\omega_0 = 1$. That is the case where the differentiated series coincides

EXHIBIT 3

ω_k (vertical axis) Ask Increases (horizontal axis)



Note: Each line is associated with a particular value of $d \in [0,1]$, in 0.1 increments.

with the original one. When $d = 1$, all weights are 0 except for $\omega_0 = 1$ and $\omega_1 = -1$. That is the standard first-order integer differentiation, which is used to derive log-price returns. Anywhere between these two cases, all weights after $\omega_0 = 1$ are negative and greater than −1. Exhibit 3 plots the weights for different orders of differentiation $d \in [0, 1]$.

Consider the series of E-mini S&P 500 log-prices. The statistic of an augmented Dickey–Fuller (ADF) test on the original series ($d = 0$) is −0.3387, at which value we cannot reject the null hypothesis of unit root with 95% confidence (the critical value is −2.8623). However, the value of an ADF statistic computed on the FracDiff series with $d = 0.4$ is −3.2733, and we can reject the null hypothesis with a confidence level in excess of 95%. Furthermore, the correlation between the original series and the FracDiff series with $d = 0.4$ is very high, around 0.995, indicating that most of the memory is still preserved. Exhibit 4 plots the ADF statistic and the correlation to the original series for various values of d. In contrast, at $d = 1$ (the standard returns), the FracDiff series has an ADF statistic of −46.9114, with a correlation to the original series of only 0.05. In other words, standard returns overdifferentiate the series, in the sense of wiping out much more memory than was necessary to achieve stationarity.

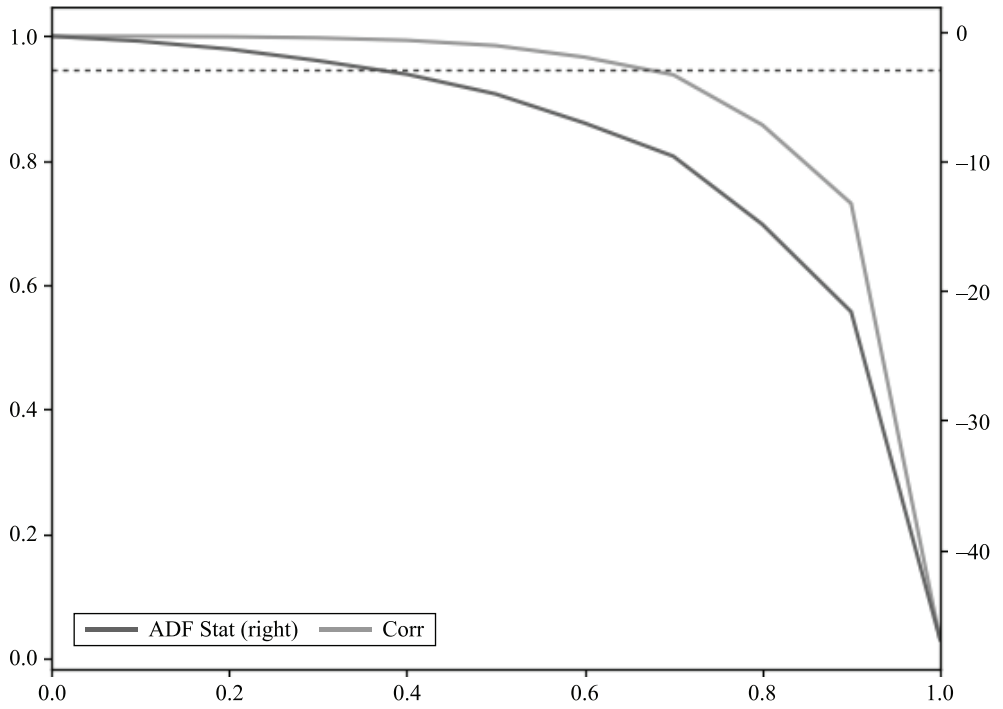
This finding is not specific to the E-mini S&P 500 log-prices. López de Prado [2018] showed that of the 87 most liquid futures contracts traded around the world, all of their log-prices achieve stationarity at $d < 0.6$, and in fact the great majority are stationary at $d < 0.3$. The conclusion is that, for decades, most empirical studies have worked with series in which memory has been unnecessarily wiped out. This is a dangerous practice because fitting a memoryless series will likely lead to a spurious pattern, a false discovery. Incidentally, this overdifferentiation of time series may explain why the efficient markets hypothesis is still so prevalent among academic circles: Without memory, series will not be predictive, and researchers may draw the false conclusion that markets are unpredictable.

PITFALL #5: FIXED-TIME HORIZON LABELING

As it relates to finance, virtually all ML papers label observations using the fixed-time horizon method. This method can be described as follows. Consider a

EXHIBIT 4

ADF Statistic as a Function of *d*, on E-Mini S&P 500 Futures Log Prices



set of features $\{X_i\}_{i=1,\ldots,I}$, drawn from some bars with index $t = 1, \ldots, T$, where $I \leq T$. An observation X_i is assigned a label $y_i \in \{-1, 0, 1\}$,

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

where τ is a predefined 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,

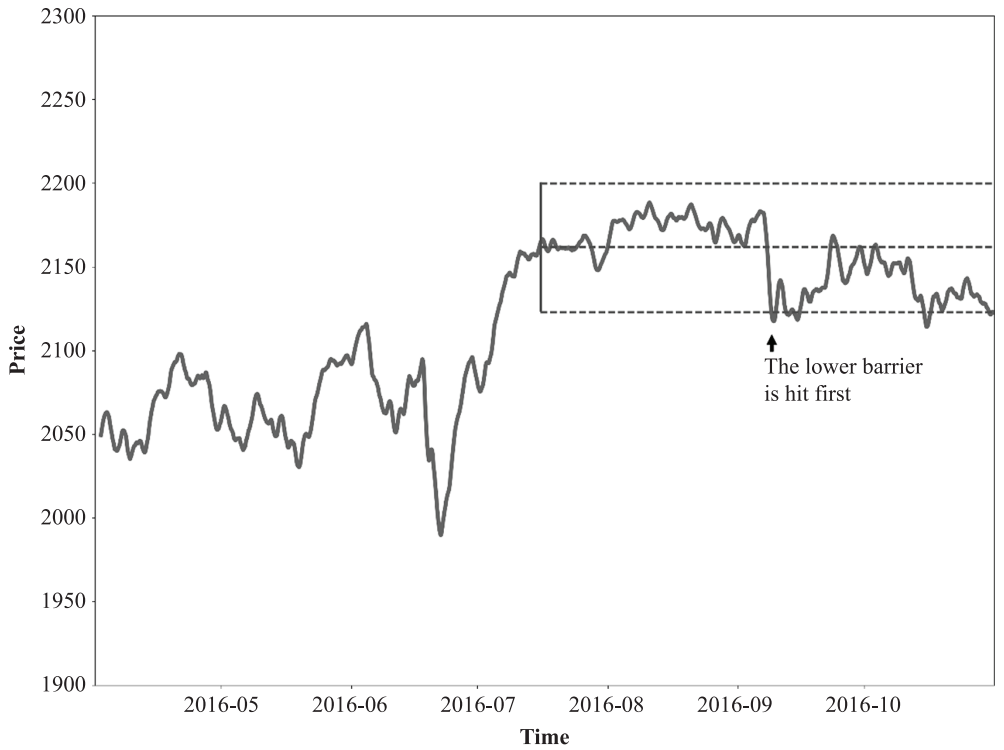
$$r_{t_{i,0},t_{i,0}+h} = \frac{p_{t_{i,0}+h}}{p_{t_{i,0}}} - 1$$

Because the literature almost always works with time bars (see Pitfall #3), h implies a fixed-time horizon. Despite its popularity, there are several arguments in favor of avoiding this approach. First, as we saw earlier, time bars do not exhibit good statistical properties. Second, 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 to $\sigma_{t_{i,0}} = 1E-2$ (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. Third, 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.

SOLUTION #5: THE TRIPLE-BARRIER METHOD

A better approach is to label observations according to the condition that triggers an exit of a position. Let us see one way to accomplish this. First, we set two horizontal barriers and one vertical barrier. The two horizontal barriers are defined by profit-taking and stop-loss limits, which are a dynamic function of estimated volatility (whether realized or implied). The third barrier is defined in terms of the number of bars elapsed since the position was taken (an activity-based,

EXHIBIT 5
The Triple-Barrier Method



not fixed-time, expiration limit). If the upper horizontal barrier is touched first, we label the observation as a 1. If the lower horizontal barrier is touched first, we label the observation as a −1. If the vertical barrier is touched first, we have two choices: the sign of the return, or a 0. I personally prefer the former as a matter of realizing a profit or loss within limits, but you should explore whether a 0 works better in your particular problems.

You may have noticed that the triple barrier method is path dependent. To label an observation, we must take into account the entire path spanning $[t_{i,0}, t_{i,0} + h]$, where h defines the vertical barrier (the expiration limit). We will denote $t_{i,1}$ the time of the first barrier touch, and the return associated with the observed feature is $r_{t_{i,0},t_{i,1}}$. For the sake of clarity, $t_{i,1} \leq t_{i,0} + h$ and the horizontal barriers are not necessarily symmetric. Exhibit 5 illustrates this labeling method.

PITFALL #6: LEARNING SIDE AND SIZE SIMULTANEOUSLY

A common error in financial ML is to build overly complex models that must learn the side and size of a position simultaneously. Let me argue why this is in general a mistake. The side decision (whether to buy or sell) is a strictly fundamental decision that has to do with the fair value of a security under a particular set of circumstances, characterized by the features matrix. However, the size decision is a risk management decision. It has to do with risk budget, funding capabilities, drawdown limits, and, very importantly, with our confidence in the side decision. Combining these two distinct decisions into a single model is unnecessary. The additional complexity involved is unwarranted.

In practice, it is often better to build two models, one to predict the side and another to predict the size of the position. The goal of the primary model is to predict the sign of the position's return. The goal of the secondary model is to predict the accuracy of the primary model's prediction. In other words, the secondary model does not attempt to predict the market but to learn from the weaknesses of the primary model. You can also think of the primary model as making trading decisions, whereas the secondary model is making risk management decisions.

There is another argument for splitting the side/size decision. Many ML models exhibit high precision (the number of true positives relative to the total number of predicted positives) and low recall (the number of true positives relative to the total number of positives). This is problematic because these models are too conservative and miss most of the opportunities. Even if these models were virtually infallible, once they enter into a drawdown, they may remain underwater for a long time as a result of their infrequent trading. It is better to develop models with a high F1-score (the harmonic average between precision and recall). This can be accomplished by splitting the side and size decisions into two models, where the secondary model applies meta-labeling.

SOLUTION #6: META-LABELING

Meta-labeling is particularly helpful when you want to achieve higher F1-scores. First, we build a primary model that achieves high recall (e.g., in predicting market rallies), even if the precision is not particularly high. Second, we correct for the low precision by labeling the bets of the primary model according to their outcome (positive or negative). The goal of these meta-labels is to increase the F1-score by filtering out the false positives, where the positives have already been identified by the primary model. Stated differently, the role of the secondary ML algorithm is to determine whether a positive from the primary (side decision) model is true or false. It is not its purpose to come up with a betting opportunity. Its goal is to determine whether we should act or pass on the opportunity that has been presented.

Meta-labeling is a very powerful tool to have in your arsenal for four additional reasons. First, ML algorithms are often criticized as black boxes. Meta-labeling allows you to build an ML system on top of a white box (such as a fundamental model founded on economic theory). This ability to transform a fundamental model into a ML model should make meta-labeling particularly useful to quantamental firms. Second, the effects of overfitting are limited when you apply meta-labeling because ML will not decide the side of your bet, only the size. There is not a single model or parameter combination that controls the overall strategy behavior. Third, by decoupling the side prediction from the size prediction, meta-labeling enables sophisticated strategy structures. For instance, consider that the features driving a rally may differ from the features driving a sell-off. In that case, you may want to develop an ML strategy exclusively for long positions, based on the buy recommendations of a primary model, and an ML strategy exclusively for short positions, based on the sell recommendations of an entirely different primary model. Fourth, achieving high accuracy on small bets and low accuracy on large bets will ruin you. Properly sizing good opportunities is as important as identifying them, so it makes sense to develop an ML algorithm solely focused on getting that critical (sizing) decision right. In my experience, meta-labeling ML models can deliver more robust and reliable outcomes than standard labeling models.

PITFALL #7: WEIGHTING OF NON–INDEPENDENT IDENTICALLY DISTRIBUTED SAMPLES

Most nonfinancial ML researchers can assume that observations are drawn from independent identically distributed (i.i.d.) processes. For example, you can obtain blood samples from a large number of patients and measure their cholesterol. Of course, various underlying common factors will shift the mean and standard deviation of the cholesterol distribution, but the samples are still independent: There is one observation per subject.

Suppose that you take those blood samples, and someone in your laboratory spills blood from each tube into the following nine tubes to their right. That is, tube 10 contains blood from patient 10 but also blood from patients 1 to 9, tube 11 contains blood from patient 11 but also blood from patients 2 to 10, and so on. Now you need to determine the features predictive of high cholesterol (diet, exercise, age, etc.), without knowing for sure the cholesterol level of each patient.

This spilled samples problem is equivalent to the challenge that we face in financial ML, in which (1) labels are decided by outcomes; (2) outcomes are decided over multiple observations; and (3) because labels overlap in time, we cannot be certain about what observed features caused an effect.

SOLUTION #7: UNIQUENESS WEIGHTING AND SEQUENTIAL BOOTSTRAPPING

Two labels, y_i and y_j, are concurrent at observation t when both are a function of at least one common return,

$r_{t-1,t} = \frac{p_t}{p_{t-1}} - 1$. We can measure the degree of uniqueness of observations as follows:

1. For each observation $t = 1, \ldots, T$, we form a binary array, $\{1_{t,i}\}_{i=1,\ldots,I}$, 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, $\overline{u}_i = \left(\sum_{t=1}^{T} u_{t,i} \right) \left(\sum_{t=1}^{T} 1_{t,i} \right)^{-1}$.
5. Sample weights can be defined in terms of the sum of the attributed returns 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. López de Prado [2018, Chapter 4] showed how this weighting scheme can be further used to bootstrap samples with low uniqueness. The general notion is that, rather than drawing all samples simultaneously, we draw the samples sequentially, where at each step we increase the probability of drawing highly unique observations and reduce the probability of drawing observations with low uniqueness. Monte Carlo experiments demonstrate that sequential bootstrapping can significantly increase the average uniqueness of samples, hence injecting more information into the model and reducing the spilled samples effect.

PITFALL #8: CROSS-VALIDATION LEAKAGE

One reason k-fold cross-validation fails in finance is because observations cannot be assumed to be drawn from an i.i.d. process. 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: (1) Because of the serial correlation, $X_t \approx X_{t+1}$, and (2) because labels are derived from overlapping data points, $Y_t \approx Y_{t+1}$. Then, placing t and $t + 1$ in different sets leaks information. When a classifier is first trained on (X_t, Y_t) and then asked to predict $\mathrm{E}[Y_{t+1}]$ based on an observed X_{t+1}, this classifier is more likely to achieve $Y_{t+1} = \mathrm{E}[Y_{t+1}]$ even if X is an irrelevant feature. In the presence of irrelevant features, leakage leads to false discoveries.

SOLUTION #8: PURGING AND EMBARGOING

One way to reduce leakage is to eliminate from the training set all observations whose labels overlapped in time with those labels included in the testing set. I call this process *purging*. Consider a label Y_j that is a function of observations in the closed range $t \in [t_{j,0}, t_{j,1}]$, $Y_j = f[[t_{j,0}, t_{j,1}]]$ (with some abuse of notation). 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 indexes $t_{j,0}$ and $t_{j,1}$, that is, $\mathrm{sgn}[r_{t_{j,0},t_{j,1}}]$. A label $Y_i = f[[t_{j,0}, t_{j,1}]]$ overlaps with Y_j if any of the three sufficient conditions is met: (1) $t_{j,0} \leq t_{i,0} \leq t_{j,1}$; (2) $t_{j,0} \leq t_{i,1} \leq t_{j,1}$; (3) $t_{i,0} \leq t_{j,0} \leq t_{j,1} \leq t_{i,1}$.

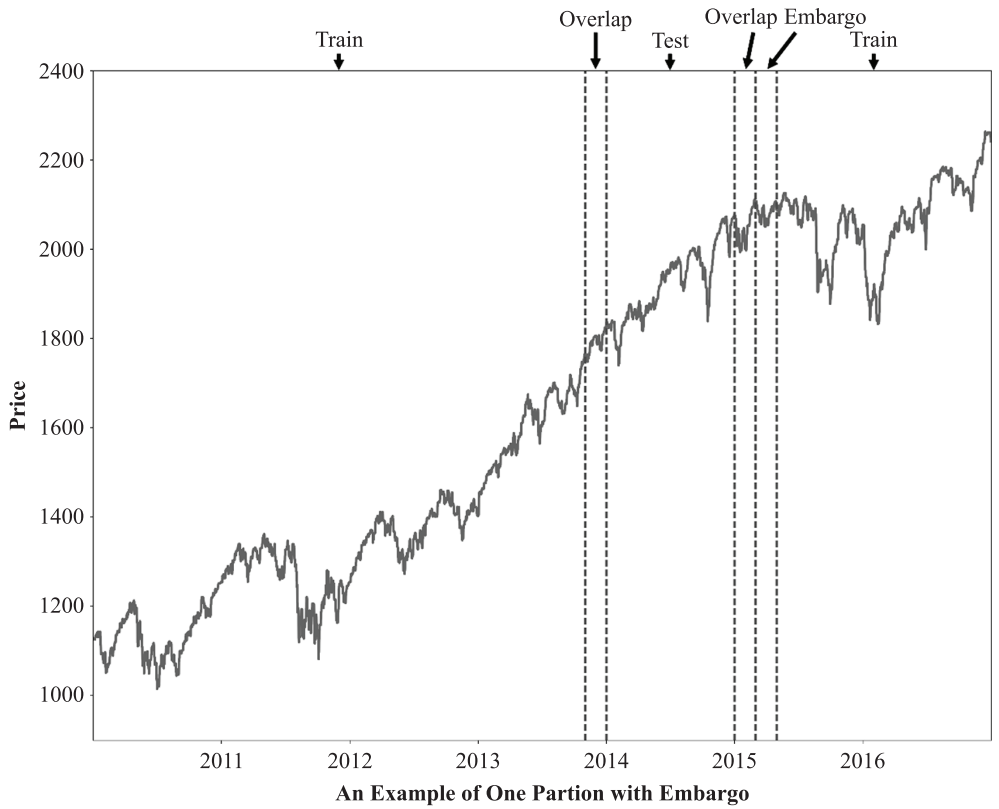
Because financial features often incorporate series that exhibit serial correlation (such as ARMA processes), we should eliminate from the training set observations that immediately follow an observation in the testing set. I call this process *embargo*. The embargo does not need to affect training observations prior to a test because training labels $Y_i = f[[t_{i,0}, t_{i,1}]]$, 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 $Y_i = f[[t_{i,0}, t_{i,1}]]$ 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 $Y_j = f[[t_{j,0}, t_{j,1} + h]]$ before purging. A small value $h \approx 0.01T$, where T is the number of bars, often suffices to prevent all leakage. Exhibit 6 illustrates how purging and embargoing would be implemented on a particular train/test split.

PITFALL #9: WF (OR HISTORICAL) BACKTESTING

The most common backtest method in the literature is the walk-forward (WF) approach. WF is an historical

EXHIBIT 6

Purging Overlap Plus Embargoing Training Examples after Test



simulation of how the strategy would have performed in the past. Each strategy decision is based on observations that predate that decision. WF enjoys two key advantages: (1) WF has a clear historical interpretation; its performance can be reconciled with paper trading. (2) History is a filtration; hence, using trailing data guarantees that the testing set is out-of-sample (no leakage), as long as purging has been properly implemented.

WF suffers from three major disadvantages: First, a single scenario is tested (the historical path), which can be easily overfit (Bailey et al. [2014]). Second, WF is not necessarily representative of future performance because results can be biased by the particular sequence of data points. Proponents of the WF method typically argue that predicting the past would lead to overly optimistic performance estimates. Yet, very often, fitting an outperforming model on the reversed sequence of observations will lead to an underperforming WF backtest. The truth is, it is as easy to overfit a WF backtest as to overfit a walk-backward backtest, and the fact that changing the sequence of observations yields inconsistent outcomes is evidence of that overfitting.

If proponents of WF were right, we should observe that walk-backward backtests systematically outperform their WF counterparts. That is not the case; hence, the main argument in favor of WF is rather weak. To make this second disadvantage clearer, suppose an equity strategy that is backtested with a WF on S&P 500 data, starting January 1, 2007. Until March 15, 2009, the mix of rallies and sell-offs will train the strategy to be market neutral, with low confidence on every position. After that, the long rally will dominate the dataset, and by January 1, 2017, buy forecasts will prevail over sell forecasts. Performance would be very different if we played the information backward from January 1, 2017, to January 1, 2007 (a long rally followed by a sharp sell-off). By exploiting a particular sequence, a strategy selected by WF may set us up for a debacle. The third disadvantage of WF is that the initial decisions are made on a smaller portion of the total sample. Even if a

EXHIBIT 7

Paths Generated for φ [6.2] = 5

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	Paths
G1	x	x	x	x	x											5
G2	x					x	x	x	x							5
G3		x				x				x	x	x				5
G4			x				x			x			x	x		5
G5				x				x			x		x		x	5
G6					x				x			x		x	x	5

EXHIBIT 8

Assignment of Testing Groups to Each of the Five Paths

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	Paths
G1	1	2	3	4	5											5
G2	1					2	3	4	5							5
G3		1				2				3	4	5				5
G4			1				2			3			4	5		5
G5				1				2			3		4		5	5
G6					1				2			3		4	5	5

warm-up period is set, most of the information is used by only a small portion of the decisions.

SOLUTION #9: COMBINATORIAL PURGED CROSS-VALIDATION

The three pitfalls of WF can be addressed by simulating a large number of scenarios in which each scenario provides us with a backtest path. This in turn will allow us to fully use the data and avoid warm-up periods. One way to achieve this is by generating thousands of train/test splits so that every part of the series is tested multiple times (not just once). Let us outline how the combinatorial purged cross-validation (CPCV) method works.

Consider T observations partitioned into N groups without shuffling, where groups $n = 1, \ldots, N-1$ are of size $\lfloor T/N \rfloor$, the Nth group is of size $T - \lfloor T/N \rfloor (N-1)$, and $\lfloor . \rfloor$ is the floor or integer function. For a testing set of size k groups, the number of possible training/testing splits is

$$\binom{N}{N-k} = \frac{\prod_{i=0}^{k-1}(N-i)}{k!}$$

Because each combination involves k tested groups, the total number of tested groups is $k\binom{N}{N-k}$. Furthermore, because we have computed all possible combinations, these tested groups are uniformly distributed across all N (each group belongs to the same number of training and testing sets). The implication is that from k-sized testing sets on N groups we can backtest a total number of paths $\varphi[N, K]$,

$$\varphi[N,k] = \frac{k}{N}\binom{N}{N-k} = \frac{\prod_{i=1}^{k-1}(N-i)}{(k-1)!}$$

Exhibit 7 illustrates the composition of train/test splits for $N = 6$ and $k = 2$. There are $\binom{6}{4} = 15$ splits, indexed as $S1, \ldots, S15$. For each split, the figure marks with a cross (x) the groups included in the testing set and leaves unmarked the groups that form the training set. Each group forms part of $\varphi[6, 2] = 5$ testing sets;

therefore, this train/test split scheme allows us to compute five backtest paths.

Exhibit 8 shows the assignment of each tested group to one backtest path. For example, path 1 is the result of combining the forecasts from $(G1, S1)$,$(G2, S1)$,$(G3, S2)$, $(G4, S3)$,$(G5, S4)$, and $(G6, S5)$; path 2 is the result of combining forecasts from $(G1, S2)$,$(G2, S6)$,$(G3, S6)$, $(G4, S7)$,$(G5, S8)$, and $(G6, S9)$; and so on.

In this example, we have generated only five paths; however, CPCV allows us to generate thousands of paths on a sufficiently long series. The number of paths $\varphi[N, k]$ increases with $N \to T$ and with $k \to N/2$. A key advantage of CPCV is that it allows us to derive a distribution of Sharpe ratios, as opposed to a single (likely overfit) Sharpe ratio estimate.

PITFALL #10: BACKTEST OVERFITTING

Given a sample of i.i.d. random variables, $x_i \sim Z$, $i = 1, \ldots, I$, where Z is the standard normal distribution, the expected maximum of that sample can be approximated as

$$\mathrm{E}[\max\{x_i\}_{i=1,\ldots,I}] \approx (1-\gamma)Z^{-1}\left[1-\frac{1}{I}\right]+\gamma Z^{-1}\left[1-\frac{1}{I}e^{-1}\right] \leq \sqrt{2\log[I]}$$

where $Z^{-1}[.]$ is the inverse of the CDF of Z, $\gamma \approx 0.5772156649 \ldots$ is the Euler–Mascheroni constant, and $I >> 1$ (see Bailey et al. [2014] for a proof). Now suppose that a researcher backtests I independent strategies on an instrument that behaves like a martingale, with Sharpe ratios $\{y_i\}_{i=1,\ldots,I}$, $\mathrm{E}[y_i]=0$, $\sigma^2[y_i] > 0$, and $\frac{y_i}{\sigma[y_i]} \sim Z$. Even though the true Sharpe ratio is zero, we expect to find one strategy with a Sharpe ratio of

$$\mathrm{E}[\max\{y_i\}_{i=1,\ldots,I}] = \mathrm{E}[\max\{x_i\}_{i=1,\ldots,I}]\sigma[y_i]$$

WF backtests exhibit high variance, $\sigma[y_i] >> 0$, for at least one reason: A large portion of the decisions are based on a small portion of the dataset. A few observations will have a large weight on the Sharpe ratio. Using a warm-up period will reduce the backtest length, which may contribute to making the variance even higher. WF's high variance leads to false discoveries because researchers will select the backtest with the maximum estimated Sharpe ratio, even if the true Sharpe ratio is zero. That is why it is imperative to control for the number of trials (I) in the context of WF backtesting. Without this information, it is not possible to determine the familywise error rate, false discovery rate, probability of backtest overfitting, or similar.

SOLUTION #10: THE DEFLATED SHARPE RATIO

The probabilistic Sharpe ratio (PSR) provides an adjusted estimate of the Sharpe ratio by removing the inflationary effect caused by short series with skewed and/or fat-tailed returns. Given a user-defined rejection threshold SR^* and an observed Sharpe ratio $\widehat{SR}$, PSR estimates the probability that $\widehat{SR}$ is greater than a hypothetical SR^*. Following Bailey and López de Prado [2012], PSR can be estimated as

$$\widehat{PSR}[SR^*] = Z\left[\frac{(\widehat{SR}-SR^*)\sqrt{T-1}}{\sqrt{1-\hat{\gamma}_3\widehat{SR}+\frac{\hat{\gamma}_4-1}{4}\widehat{SR}^2}}\right]$$

where $Z[.]$ is the CDF of the standard normal distribution, T is the number of observed returns, $\hat{\gamma}_3$ is the skewness of the returns, and $\hat{\gamma}_4$ is the kurtosis of the returns ($\hat{\gamma}_4 = 3$ for Gaussian returns). For a given SR^*, $\widehat{PSR}$ increases with greater $\widehat{SR}$ (in the original sampling frequency, i.e., nonannualized), longer track records (T), or positively skewed returns ($\hat{\gamma}_3$), but it decreases with fatter tails ($\hat{\gamma}_4$).

The deflated Sharpe ratio (DSR) computes the probability that the true Sharpe ratio exceeds a rejection threshold SR^*, where that rejection threshold is adjusted to reflect the multiplicity of trials. Following Bailey and López de Prado [2014], DSR can be estimated as $\widehat{PSR}[SR^*]$, where the benchmark Sharpe ratio, SR^*, is no longer user-defined. Instead, SR^* is estimated as

$$SR^* = \sqrt{V[\{\widehat{SR}_i\}]}\left((1-\gamma)Z^{-1}\left[1-\frac{1}{I}\right]+\gamma Z^{-1}\left[1-\frac{1}{I}e^{-1}\right]\right)$$

where $V[\{\widehat{SR}_i\}]$ is the variance across the trials' estimated Sharpe ratio, I is the number of independent trials, $Z[.]$ is the cumulative distribution function of the standard normal distribution, γ is the Euler–Mascheroni constant, and $i = 1, \ldots, I$.

The rationale behind DSR is the following: Given a set of Sharpe ratio estimates, $\{\widehat{SR}_i\}$, its expected maximum is greater than zero, even if the true Sharpe ratio is zero. Under the null hypothesis that the actual Sharpe ratio is zero, H_0: $SR = 0$, we know that the expected maximum $\widehat{SR}$ can be estimated as the SR^*. Indeed, SR^* increases quickly as more independent trials are attempted (I), or the trials involve a greater variance ($V[\{\widehat{SR}_i\}]$). An unsupervised learning method for the accurate estimation of I and $V[\{\widehat{SR}_i\}]$ is explained by López de Prado and Lewis [2018].

CONCLUSIONS

Many of the most successful hedge funds in history apply ML techniques. However, ML is far from being a panacea, and a large number of funds that have attempted to join ML investing have failed because financial datasets exhibit properties that violate standard assumptions of ML applications. When ML techniques are applied to financial datasets in disregard of those properties, these techniques produce false positives. In the context of investing, the implication is that most ML funds fail to deliver the expected performance. In this article we have reviewed some of the most pervasive errors made by ML experts when they attempt to apply ML techniques to financial datasets.

ACKNOWLEDGMENT

This article is partly based on the author's book *Advances in Financial Machine Learning* (Wiley 2018).

REFERENCES

Alexander, C. *Market Models*. 1st ed. Hoboken, NJ: John Wiley & Sons, 2001.

American Statistical Association. 2016. "Ethical Guidelines for Statistical Practice." Committee on Professional Ethics of the American Statistical Association, April 2016, http://www.amstat.org/asa/files/pdfs/EthicalGuidelines.pdf.

Bailey, D., J. Borwein, M. López de Prado, and J. Zhu. 2014. "Pseudo-Mathematics and Financial Charlatanism: The Effects of Backtest Overfitting on Out-of-Sample Performance." *Notices of the American Mathematical Society* 61 (5): 458–471.

Bailey, D., and M. López de Prado. 2012. "The Sharpe Ratio Efficient Frontier." *Journal of Risk* 15 (2): 3–44.

——. 2014. "The Deflated Sharpe Ratio: Correcting for Selection Bias, Backtest Overfitting and Non-Normality." *The Journal of Portfolio Management* 40 (5): 94–107.

Calkin, N., and M. López de Prado. 2014a. "Stochastic Flow Diagrams." *Algorithmic Finance* 3 (1): 21–42.

——. 2014b. "The Topology of Macro Financial Flows: An Application of Stochastic Flow Diagrams." *Algorithmic Finance* 3 (1): 43–85.

Easley, D., M. López de Prado, and M. O'Hara. 2011. "The Volume Clock: Insights into the High Frequency Paradigm." *The Journal of Portfolio Management* 37 (2): 118–128.

——. 2012. "Flow Toxicity and Liquidity in a High Frequency World." *Review of Financial Studies* 25 (5): 1457–1493.

——. *High Frequency Trading: New Realities for Traders, Markets and Regulators,* 1st ed. London: Risk Books, 2013.

Hamilton, J. *Time Series Analysis*, 1st ed. Princeton, NJ: Princeton University Press, 1994.

López de Prado, M. 2014. "Quantitative Meta-Strategies. Practical Applications." *Institutional Investor Journals* 2 (3): 1–3.

——. *Advances in Financial Machine Learning,* 1st ed. Hoboken, NJ: Wiley, 2018.

López de Prado, M., and M. Lewis. "Detection of False Investment Strategies Using Unsupervised Learning Methods." Working paper, 2018. Available at https://ssrn.com/abstract=3167017.

Stigler, S. M. 1981. "Gauss and the Invention of Least Squares." *Annals of Statistics* 9 (3): 465–474.