1 Introduction

1.1 Motivation

To a greater extent than other mathematical disciplines, statistics is a product of its time. If Francis Galton, Karl Pearson, Ronald Fisher, and Jerzy Neyman had had access to computers, they may have created an entirely different field. Classical statistics relies on simplistic assumptions (linearity, independence), in-sample analysis, analytical solutions, and asymptotic properties partly because its founders had access to limited computing power. Today, many of these legacy methods continue to be taught at university courses and in professional certification programs, even though computational methods, such as cross-validation, ensemble estimators, regularization, bootstrapping, and Monte Carlo, deliver demonstrably better solutions. In the words of Efron and Hastie (2016, 53),

> two words explain the classic preference for parametric models: mathematical tractability. In a world of sliderules and slow mechanical arithmetic, mathematical formulation, by necessity, becomes the computational tool of choice. Our new computation-rich environment has unplugged the mathematical bottleneck, giving us a more realistic, flexible, and far-reaching body of statistical techniques.

Financial problems pose a particular challenge to those legacy methods, because economic systems exhibit a degree of complexity that is beyond the grasp of classical statistical tools (López de Prado 2019b). As a consequence, machine learning (ML) plays an increasingly important role in finance. Only a few years ago, it was rare to find ML applications outside short-term price prediction, trade execution, and setting of credit ratings. Today, it is hard to find a use case where ML is not being deployed in some form. This trend is unlikely to change, as larger data sets, greater computing power, and more efficient algorithms all conspire to unleash a golden age of financial ML. The ML revolution creates opportunities for dynamic firms and challenges for antiquated asset managers. Firms that resist this revolution will likely share Kodak's fate. One motivation of this Element is to demonstrate how modern statistical tools help address many of the deficiencies of classical techniques in the context of asset management.

Most ML algorithms were originally devised for cross-sectional data sets. This limits their direct applicability to financial problems, where modeling the time series properties of data sets is essential. My previous book, *Advances in Financial Machine Learning* (AFML; López de Prado 2018a), addressed the challenge of modeling the time series properties of financial data sets with ML algorithms, from the perspective of an academic who also happens to be a practitioner.

Machine Learning for Asset Managers is concerned with answering a different challenge: how can we use ML to build better financial theories? This is not a philosophical or rhetorical question. Whatever edge you aspire to gain in finance, it can only be justified in terms of someone else making a systematic mistake from which you benefit.[1] Without a testable theory that explains your edge, the odds are that you do not have an edge at all. A historical simulation of an investment strategy's performance (backtest) is not a theory; it is a (likely unrealistic) simulation of a past that never happened (you did not deploy that strategy years ago; that is why you are backtesting it!). Only a theory can pin down the clear cause–effect mechanism that allows you to extract profits against the collective wisdom of the crowds – a testable theory that explains factual evidence as well as counterfactual cases (x implies y, and the absence of y implies the absence of x). Asset managers should focus their efforts on researching theories, not backtesting trading rules. ML is a powerful tool for building financial theories, and the main goal of this Element is to introduce you to essential techniques that you will need in your endeavor.

1.2 Theory Matters

A black swan is typically defined as an extreme event that has not been observed before. Someone once told me that quantitative investment strategies are useless. Puzzled, I asked why. He replied, "Because the future is filled with black swans, and since historical data sets by definition cannot contain never-seen-before events, ML algorithms cannot be trained to predict them." I counter-argued that, in many cases, black swans have been predicted.

Let me explain this apparent paradox with an anecdote. Back in the year 2010, I was head of high-frequency futures at a large US hedge fund. On May 6, we were running our liquidity provision algorithms as usual, when around 12:30 ET, many of them started to flatten their positions automatically. We did not interfere or override the systems, so within minutes, our market exposure became very small. This system behavior had never happened to us before. My team and I were conducting a forensic analysis of what had caused the systems to shut themselves down when, at around 14:30 ET, we saw the S&P 500 plunge, within minutes, almost 10% relative to the open. Shortly after, the systems started to buy aggressively, profiting from a 5% rally into the market close. The press dubbed this black swan the "flash crash." We were twice surprised by this episode: first, we could not understand how our systems

[1] This is also true in the context of factor investing, where the systematic mistake can be explained in terms of behavioral bias, mismatched investment horizons, risk tolerance, regulatory constraints, and other variables informing investors' decisions.

predicted an event that we, the developers, did not anticipate; second, we could not understand why our systems started to buy shortly after the market bottomed.

About five months later, an official investigation found that the crash was likely caused by an order to sell 75,000 E-mini S&P 500 futures contracts at a high participation rate (CFTC 2010). That large order contributed to a persistent imbalance in the order flow, making it very difficult for market makers to flip their inventory without incurring losses. This toxic order flow triggered stop-out limits across market makers, who ceased to provide liquidity. Market makers became aggressive liquidity takers, and without anyone remaining on the bid, the market inevitably collapsed (Easley et al. 2011).

We could not have forecasted the flash crash by watching CNBC or reading the *Wall Street Journal*. To most observers, the flash crash was indeed an unpredictable black swan. However, the underlying causes of the flash crash are very common. Order flow is almost never perfectly balanced. In fact, imbalanced order flow is the norm, with various degrees of persistency (e.g., measured in terms of serial correlation). Our systems had been trained to reduce positions under extreme conditions of order flow imbalance. In doing so, they were trained to avoid the conditions that shortly after caused the black swan. Once the market collapsed, our systems recognized that the opportunity to buy at a 10% discount offset previous concerns from extreme order flow imbalance, and they took long positions until the close. This experience illustrates the two most important lessons contained in this Element.

1.2.1 Lesson 1: You Need a Theory

Contrary to popular belief, backtesting is not a research tool. Backtests can never prove that a strategy is a true positive, and they may only provide evidence that a strategy is a false positive. Never develop a strategy solely through backtests. Strategies must be supported by theory, not historical simulations. Your theories must be general enough to explain particular cases, even if those cases are black swans. The existence of black holes was predicted by the theory of general relativity more than five decades before the first black hole was observed. In the above story, our market microstructure theory (which later on became known as the VPIN theory; see Easley et al. 2011b) helped us predict and profit from a black swan. Not only that, but our theoretical work also contributed to the market's bounce back (my colleagues used to joke that we helped put the "flash" into the "flash crash"). This Element contains some of the tools you need to discover your own theories.

1.2.2 Lesson 2: ML Helps Discover Theories

Consider the following approach to discovering new financial theories. First, you apply ML tools to uncover the hidden variables involved in a complex

phenomenon. These are the ingredients that the theory must incorporate in order to make successful forecasts. The ML tools have identified these ingredients; however, they do not directly inform you about the exact equation that binds the ingredients together. Second, we formulate a theory that connects these ingredients through a structural statement. This structural statement is essentially a system of equations that hypothesizes a particular cause–effect mechanism. Third, the theory has a wide range of testable implications that go beyond the observations predicted by the ML tools in the first step.[2] A successful theory will predict events out-of-sample. Moreover, it will explain not only positives (x causes y) but also negatives (the absence of y is due to the absence of x).

In the above discovery process, ML plays the key role of decoupling the search for variables from the search for specification. Economic theories are often criticized for being based on "facts with unknown truth value" (Romer 2016) and "generally phony" assumptions (Solow 2010). Considering the complexity of modern financial systems, it is unlikely that a researcher will be able to uncover the ingredients of a theory by visual inspection of the data or by running a few regressions. Classical statistical methods do not allow this decoupling of the two searches.

Once the theory has been tested, it stands on its own feet. In this way, the theory, not the ML algorithm, makes the predictions. In the above anecdote, the theory, not an online forecast produced by an autonomous ML algorithm, shut the position down. The forecast was theoretically sound, and it was not based on some undefined pattern. It is true that the theory could not have been discovered without the help of ML techniques, but once the theory was discovered, the ML algorithm played no role in the decision to close the positions two hours prior to the flash crash. The most insightful use of ML in finance is for discovering theories. You may use ML successfully for making financial forecasts; however, that is not necessarily the best scientific use of this technology (particularly if your goal is to develop high-capacity investment strategies).

1.3 How Scientists Use ML

An ML algorithm learns complex patterns in a high-dimensional space with little human guidance on model specification. That ML models need not be specified by the researcher has led many to, erroneously, conclude that ML must

[2] A theory can be tested with more powerful tools than backtests. For instance, we could investigate which market makers lost money during the flash crash. Did they monitor for order flow imbalance? Did market makers that monitor for order flow imbalance fare better? Can we find evidence of their earlier retreat in the FIX messages of that day? A historical simulation of a trading rule cannot give us this level of insight.

be a black box. In that view, ML is merely an "oracle,"[3] a prediction machine from which no understanding can be extracted. The black box view of ML is a misconception. It is fueled by popular industrial applications of ML, where the search for better predictions outweighs the need for theoretical understanding. A review of recent scientific breakthroughs reveals radically different uses of ML in science, including the following:

1 **Existence:** ML has been deployed to evaluate the plausibility of a theory across all scientific fields, even beyond the empirical sciences. Notably, ML algorithms have helped make mathematical discoveries. ML algorithms cannot prove a theorem, however they can point to the existence of an undiscovered theorem, which can then be conjectured and eventually proved. In other words, if something can be predicted, there is hope that a mechanism can be uncovered (Gryak et al., forthcoming).
2 **Importance:** ML algorithms can determine the relative informational content of explanatory variables (features, in ML parlance) for explanatory and/or predictive purposes (Liu 2004). For example, the mean-decrease accuracy (MDA) method follows these steps: (1) Fit a ML algorithm on a particular data set; (2) derive the out-of-sample cross-validated accuracy; (3) repeat step (2) after shuffling the time series of individual features or combinations of features; (4) compute the decay in accuracy between (2) and (3). Shuffling the time series of an important feature will cause a significant decay in accuracy. Thus, although MDA does not uncover the underlying mechanism, it discovers the variables that should be part of the theory.
3 **Causation:** ML algorithms are often utilized to evaluate causal inference following these steps: (1) Fit a ML algorithm on historical data to predict outcomes, absent of an effect. This model is nontheoretical, and it is purely driven by data (like an oracle); (2) collect observations of outcomes under the presence of the effect; (3) use the ML algorithm fit in (1) to predict the observation collected in (2). The prediction error can be largely attributed to the effect, and a theory of causation can be proposed (Varian 2014; Athey 2015).
4 **Reductionist:** ML techniques are essential for the visualization of large, high-dimensional, complex data sets. For example, manifold learning algorithms can cluster a large number of observations into a reduced subset of peer groups, whose differentiating properties can then be analyzed (Schlecht et al. 2008).

[3] Here we use a common definition of oracle in complexity theory: a black box that is able to produce a solution for any instance of a given computational problem.

5 **Retriever:** ML is used to scan through big data in search of patterns that humans failed to recognize. For instance, every night ML algorithms are fed millions of images in search of supernovae. Once they find one image with a high probability of containing a supernova, expensive telescopes can be pointed to a particular region in the universe, where humans will scrutinize the data (Lochner et al. 2016). A second example is outlier detection. Finding outliers is a prediction problem rather than an explanation problem. A ML algorithm can detect an anomalous observation, based on the complex structure it has found in the data, even if that structure is not explained to us (Hodge and Austin 2004).

Rather than replacing theories, ML plays the critical role of helping scientists form theories based on rich empirical evidence. Likewise, ML opens the opportunity for economists to apply powerful data science tools toward the development of sound theories.

1.4 Two Types of Overfitting

The dark side of ML's flexibility is that, in inexperienced hands, these algorithms can easily overfit the data. The primary symptom of overfitting is a divergence between a model's in-sample and out-of-sample performance (known as the generalization error). We can distinguish between two types of overfitting: the overfitting that occurs on the train set, and the overfitting that occurs on the test set. Figure 1.1 summarizes how ML deals with both kinds of overfitting.

1.4.1 Train Set Overfitting

Train set overfitting results from choosing a specification that is so flexible that it explains not only the signal, but also the noise. The problem with confounding signal with noise is that noise is, by definition, unpredictable. An overfit model will produce wrong predictions with an unwarranted confidence, which in turn will lead to poor performance out-of-sample (or even in a pseudo-out-of-sample, like in a backtest).

ML researchers are keenly aware of this problem, which they address in three complementary ways. The first approach to correct for train set overfitting is evaluating the generalization error, through resampling techniques (such as cross-validation) and Monte Carlo methods. Appendix A describes these techniques and methods in greater detail. The second approach to reduce train set overfitting is regularization methods, which prevent model complexity unless it can be justified in terms of greater explanatory power. Model

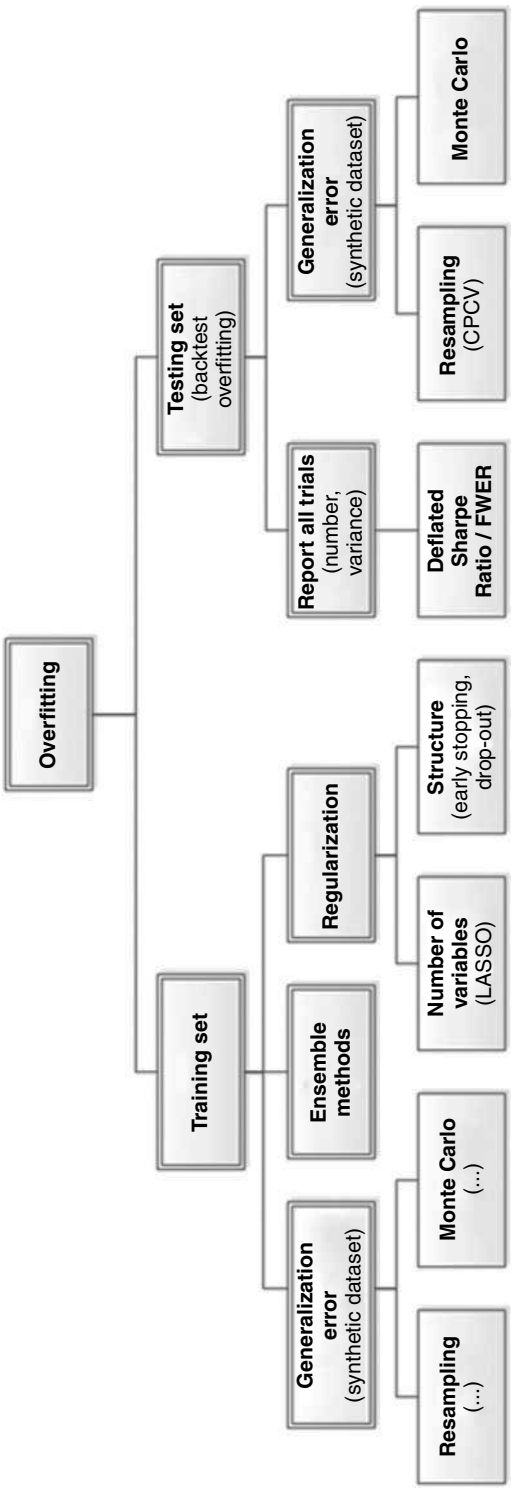

Figure 1.1 Solutions to two kinds of overfitting.

parsimony can be enforced by limiting the number of parameters (e.g., LASSO) or restricting the model's structure (e.g., early stopping). The third approach to address train set overfitting is ensemble techniques, which reduce the variance of the error by combining the forecasts of a collection of estimators. For example, we can control the risk of overfitting a random forest on a train set in at least three ways: (1) cross-validating the forecasts; (2) limiting the depth of each tree; and (3) adding more trees.

In summary, a backtest may hint at the occurrence of train set overfitting, which can be remedied using the above approaches. Unfortunately, backtests are powerless against the second type of overfitting, as explained next.

1.4.2 Test Set Overfitting

Imagine that a friend claims to have a technique to predict the winning ticket at the next lottery. His technique is not exact, so he must buy more than one ticket. Of course, if he buys all of the tickets, it is no surprise that he will win. How many tickets would you allow him to buy before concluding that his method is useless? To evaluate the accuracy of his technique, you should adjust for the fact that he has bought multiple tickets. Likewise, researchers running multiple statistical tests on the same data set are more likely to make a false discovery. By applying the same test on the same data set multiple times, it is guaranteed that eventually a researcher will make a false discovery. This selection bias comes from fitting the model to perform well on the test set, not the train set.

Another example of test set overfitting occurs when a researcher backtests a strategy and she tweaks it until the output achieves a target performance. That backtest–tweak–backtest cycle is a futile exercise that will inevitably end with an overfit strategy (a false positive). Instead, the researcher should have spent her time investigating how the research process misled her into backtesting a false strategy. In other words, a poorly performing backtest is an opportunity to fix the research process, not an opportunity to fix a particular investment strategy.

Most published discoveries in finance are likely false, due to test set overfitting. ML did not cause the current crisis in financial research (Harvey et al. 2016). That crisis was caused by the widespread misuse of classical statistical methods in finance, and *p-hacking* in particular. ML can help deal with the problem of test set overfitting, in three ways. First, we can keep track of how many independent tests a researcher has run, to evaluate the probability that at least one of the outcomes is a false discovery (known as familywise error rate, or FWER). The deflated Sharpe ratio (Bailey and López de Prado 2014) follows

a similar approach in the context of backtesting, as explained in Section 8. It is the equivalent to controlling for the number of lottery tickets that your friend bought. Second, while it is easy to overfit a model to one test set, it is hard to overfit a model to thousands of test sets for each security. Those thousands of test sets can be generated by resampling combinatorial splits of train and test sets. This is the approached followed by the combinatorial purged cross-validation method, or CPCV (AFML, chapter 12). Third, we can use historical series to estimate the underlying data-generating process, and sample synthetic data sets that match the statistical properties observed in history. Monte Carlo methods are particularly powerful at producing synthetic data sets that match the statistical properties of a historical series. The conclusions from these tests are conditional to the representativeness of the estimated data-generating process (AFML, chapter 13). The main advantage of this approach is that those conclusions are not connected to a particular (observed) realization of the data-generating process but to an entire distribution of random realizations. Following with our example, this is equivalent to replicating the lottery game and repeating it many times, so that we can rule luck out.

In summary, there are multiple practical solutions to the problem of train set and test set overfitting. These solutions are neither infallible nor incompatible, and my advice is that you apply all of them. At the same time, I must insist that no backtest can replace a theory, for at least two reasons: (1) backtests cannot simulate black swans – only theories have the breadth and depth needed to consider the never-before-seen occurrences; (2) backtests may insinuate that a strategy is profitable, but they do not tell us why. They are not a controlled experiment. Only a theory can state the cause–effect mechanism, and formulate a wide range of predictions and implications that can be independently tested for facts and counterfacts. Some of these implications may even be testable outside the realm of investing. For example, the VPIN theory predicted that market makers would suffer stop-outs under persistent order flow imbalance. Beyond testing whether order flow imbalance causes a reduction in liquidity, researchers can also test whether market makers suffered losses during the flash crash (hint: they did). This latter test can be conducted by reviewing financial statements, independently from the evidence contained in exchange records of prices and quotes.

1.5 Outline

This Element offers asset managers a step-by-step guide to building financial theories with the help of ML methods. To that objective, each section uses what we have learned in the previous ones. Each section (except for this introduction)

contains an empirical analysis, where the methods explained are put to the test in Monte Carlo experiments.

The first step in building a theory is to collect data that illustrate how some variables relate to each other. In financial settings, those data often take the form of a covariance matrix. We use covariance matrices to run regressions, optimize portfolios, manage risks, search for linkages, etc. However, financial covariance matrices are notoriously noisy. A relatively small percentage of the information they contain is signal, which is systematically suppressed by arbitrage forces. Section 2 explains how to denoise a covariance matrix without giving up the little signal it contains. Most of the discussion centers on random matrix theory, but at the core of the solution sits an ML technique: the kernel density estimator.

Many research questions involve the notion of similarity or distance. For example, we may be interested in understanding how *closely* related two variables are. Denoised covariance matrices can be very useful for deriving distance metrics from linear relationships. Modeling nonlinear relationships requires more advanced concepts. Section 3 provides an information-theoretic framework for extracting complex signals from noisy data. In particular, it allows us to define distance metrics with minimal assumptions regarding the underlying variables that characterize the metric space. These distance metrics can be thought of as a nonlinear generalization of the notion of correlation.

One of the applications of distance matrices is to study whether some variables are more closely related among themselves than to the rest, hence forming clusters. Clustering has a wide range of applications across finance, like in asset class taxonomy, portfolio construction, dimensionality reduction, or modeling networks of agents. A general problem in clustering is finding the optimal number of clusters. Section 4 introduces the ONC algorithm, which provides a general solution to this problem. Various use cases for this algorithm are presented throughout this Element.

Clustering is an unsupervised learning problem. Before we can delve into supervised learning problems, we need to assess ways of labeling financial data. The effectiveness of a supervised ML algorithm greatly depends on the kind of problem we attempt to solve. For example, it may be harder to forecast tomorrow's S&P 500 return than the sign of its next 5% move. Different features are appropriate for different types of labels. Researchers should consider carefully what labeling method they apply on their data. Section 5 discusses the merits of various alternatives.

AFML warned readers that backtesting is not a research tool. Feature importance is. A backtest cannot help us develop an economic or financial theory.