Avoiding the Pitfalls on Stock Market: Challenges and Solutions in Developing Quantitative Strategies

Marco Bergianti¹, Nicola Cioffo¹, Francesco Del Buono¹, Matteo Paganelli¹ and Angelo Porrello²

¹Cristail, Italy
²University of Modena and Reggio Emilia, Italy

Abstract
Quantitative stock trading based on Machine Learning (ML) and Deep Learning (DL) has gained great attention in recent years thanks to the ever-increasing availability of financial data and the ability of this technology to analyze the complex dynamics of the stock market. Despite the plethora of approaches present in literature, a large gap exists between the solutions produced by the scientific community and the practices adopted in real-world systems. Most of these works in fact lack a practical vision of the problem and ignore the main issues afflicting fintech practitioners. To fill such a gap, we provide a systematic review of the main dangers affecting the development of an ML-DL pipeline in the financial domain. They include managing the stochastic and non-stationary characteristics of stock data, various types of bias, overfitting of models and devising impartial valuation methods. Finally, we present possible solutions to these critical issues.

Keywords
Financial Markets, Quantitative Trading Strategies, Machine Learning, Deep Learning, Bias

1. Introduction
Stock selection is a crucial task in investment management, which has undergone a massive renewal in recent years. Given the ever-increasing availability of financial data, the traditional statistical techniques for stock selection have been gradually replaced by the most modern Machine Learning (ML) and Deep Learning (DL) methodologies, by virtue of their effectiveness in identifying hidden patterns with high predictive power.

This technology, when applied in the financial domain, is mainly used to predict stock prices, their trends (i.e., positive or negative depending on whether stock prices are expected to increase or decrease) or directly the most profitable stocks. In the first two scenarios, regressors and classifiers are respectively employed to predict the future behavior of the stocks, while in the last case the model is trained to learn a ranking function that sorts stocks in descending order by expected profit. The outputs of these models are then exploited to select the top-k most profitable stocks and to build trading strategies.

In literature, a large variety of financial models have been proposed to solve these tasks. They can be classified into methods based on technical analysis (TA) and approaches based on fundamental analysis (FA) [1].

In more detail, this paper will dive through the main macro-steps of a typical ML-DL pipeline, namely data preparation, featurization, modeling and evaluation. For
3. Data Preparation

Preparing data for financial models is a crucial task as it requires handling incomplete and inaccurate data with different forms of bias. Indeed, biased data can lead to the development of ineffective trading strategies that underperform in the real market.

3.1. Outliers and missing values

Financial data frequently contains stocks that trade intermittently and outliers (e.g., price values that deviate strongly from average behavior), which can reveal abnormal patterns (e.g., abnormal returns). Managing these anomalies is much more pressing in the financial domain than in any other field as financial decisions are often critical and profit-driven, i.e., even small errors can result in significant losses. Furthermore, they can negatively affect the training of ML/DL models, which acquire a distorted knowledge of the task. A possible solution to the first problem is to consider only the stocks that have been traded on more than a certain percentage of trading days (e.g., 98%), while the standard method to deal with outliers is to clip values within a specific range [12].

3.2. Look-ahead bias

Look-ahead bias occurs when a model uses information that would not have been available at inference time [8].

A generic approach to solve this problem is to implement out-of-sample testing, which involves dividing the data into two parts: one for model construction and one for validation. The model is trained on the first part of the data and then tested on the second part of the data. This approach can help avoid overfitting the data and that its performance is more accurately estimated.

Despite the use of this technique, look-ahead bias may still emerge when processing adjusted price data and fundamental data. Adjusted prices, for example, are constantly updated based on the occurrence of a split or the payment of dividends. When such events occur, all past time series is corrected accordingly. For example, when a 2-for-1 stock split occurs, all prices before that date are
halved. As a consequence of this, adjusted prices implicitly store information about future events and should be used with caution. To mitigate this problem, the yield series is preferred rather than the original series. It operates on percentage differences rather than on absolute values and is not affected by the bias produced by such corrections.

When fundamental data is processed, instead, it is necessary to pay attention to its publication process. These documents are written on a certain date and subsequently corrected without updating the filling date, implicitly indicating that the new information was already known at the initial writing time of the document. Not considering this aspect means including future information in the historical data, and results in inflated performance.

3.3. Survival bias
Survivorship bias occurs when the data used to train and test a model only includes the stocks that have survived until the present time, hence ignoring that some companies went bankrupt and securities were delisted. This bias can result in an overestimation of the performance of the strategies as they ignore the stocks that have gone bankrupt or delisted [8, 13, 14]. Various solutions have been proposed in the literature to address this bias, such as including delisted securities in the analysis [15] or applying a survivorship bias correction method, which involves adjusting the returns of surviving securities to account for the returns of the delisted securities.

4. Featurization
The data preparation phase is typically followed by a featurization phase, which aims at transforming the raw data in order to 1) highlight expressive patterns for the stock selection task and 2) obtain better statistical properties that facilitate processing through ML/DL. This procedure is mainly applied to raw stock price series, which exhibit unwanted properties such as non-homogeneity (i.e., values arrive with irregular frequency) and non-stationarity (i.e., their statistical properties vary over time).

In this section, we present some solutions to these problems, distinguishing between solutions for the input (i.e., feature space) and the output (i.e., label space).

4.1. Input
A very popular category of stock selection approaches is based on technical analysis, which directly elaborates on numerical features like past prices and macroeconomic indicators. This type of data is affected by several problematic conditions that must be managed appropriately to create effective trading strategies.

4.1.1. Inhomogeneous series
In literature stock price series are typically time-indexed, i.e., their values are sampled at fixed time intervals. It represents the most intuitive choice as it is consistent with sunlight cycles. Unfortunately, markets are operated by algorithms that trade with limited human supervision, for which CPU processing cycles are much more relevant than chronological intervals [16]. As a consequence, sampling information on a time basis would result in oversampling during low-activity periods and undersampling during high-activity periods. Furthermore time-sampled series often exhibit poor statistical properties, like serial correlation, heteroscedasticity, and non-normality of returns. To alleviate this problem, alternative forms of sampling have been proposed, such as volume bars that collect information whenever a certain amount of stock units have been traded, or dollar bars that sample data every time a pre-defined market value is exchanged.

4.1.2. Non-stationarity
Another undesired property of the raw stock price series is non-stationarity [17, 18], i.e., when its statistical properties vary over time. This prevents the direct application of inferential analysis as they operate exclusively on invariant processes. To circumvent this problem, the most adopted solution is to transform the raw price series into a yield series, where the absolute values of the prices are replaced by percentage variations. Although this transformation makes the series stationary, its drawback is that it removes memory from the data (i.e., removes correlations between past and future observations), which is the main bias for the model’s predictive power. Recent featurization methodologies based on fractionally differentiated features have been explored to obtain an effective trade-off between stationarity and memory [8].

4.2. Output
Parallel to the input featurization, the label space must be transformed coherently with the type of task to be solved (i.e., classification or regression).

4.2.1. Class unbalanced distribution
In a classification scenario, observations are typically labeled based on whether the return is positive or negative. However, this may produce unbalanced classes, as during market booms the probability of a positive return is much higher, and during market crashes they are lower [19]. This unbalanced distribution can introduce a bias in the model training by favoring the more frequent classes over the rarer ones. To avoid this condition, in [20] an asymmetric threshold assignment is used to balance the
Table 1
   Toy example derived from [4] showing that accurate regressors/classifiers (e.g., R1, C1) may be less profitable than other under-optimized methods (e.g., R2, C2).

<table>
<thead>
<tr>
<th>Stocks</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>Performance</th>
<th>Profits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns</td>
<td>+30</td>
<td>-10</td>
<td>+20</td>
<td>+5</td>
<td>-30</td>
<td>MAE</td>
<td>Top-1 Stock</td>
</tr>
<tr>
<td>R1</td>
<td>+20</td>
<td>-5</td>
<td>+25</td>
<td>+10</td>
<td>-20</td>
<td>7</td>
<td>MAE</td>
</tr>
<tr>
<td>R2</td>
<td>+20</td>
<td>-15</td>
<td>+10</td>
<td>+15</td>
<td>-10</td>
<td>11</td>
<td>MAE</td>
</tr>
<tr>
<td>C1</td>
<td>7.60</td>
<td>0.60</td>
<td>7.70</td>
<td>0.55</td>
<td>0.53</td>
<td>60%</td>
<td>Acc.</td>
</tr>
<tr>
<td>C2</td>
<td>7.60</td>
<td>0.60</td>
<td>7.60</td>
<td>0.60</td>
<td>0.60</td>
<td>60%</td>
<td>Acc.</td>
</tr>
</tbody>
</table>

5. Modeling

Given stock features and related labels, the next step is to apply supervised approaches to learn hidden patterns in past data and acquire predictive capabilities on future data. Several challenges afflict the design of ML/DL models in the financial domain, such as the management of the stochastic nature of data (mainly in price series), the exploitation of correlations between stocks and the correct definition of the model optimization function (e.g., identify the most profitable stocks).

5.1. Stochasticity

Stock data have a chaotic and noisy nature: they are largely driven by new information and result in a random-walk pattern [20]. This random component can negatively impact the training process. Traditional supervised techniques are in fact designed to operate on clean data and are not capable of handling uncertain data. This has motivated an intense effort in the area of deep learning, leading to several solutions over the last few years. Among these, three categories of methods have been explored: 1) the adoption of ad-hoc loss functions, 2) the exploitation of adversarial training procedures, and 3) the construction of intrinsically probabilistic models.

The intuition behind the first category is to model the output in probabilistic terms, estimating a probability distribution and not relying on punctual targets. Quantile loss [21] and Gaussian loss [22] represent the main objective functions used in this category of methods. Adversarial training approaches instead try to manage the stochasticity by training the model to produce similar outputs for different variations of the same target input [18]. Finally, instead of using only deterministic features, generative models incorporate inherently probabilistic components. Variational auto-encoders (VAE) [23] are the best-known example of this component and several stock selection approaches rely on them [20, 24].

5.2. Covariates

The stock market is also characterized by significant forms of correlation between stocks, e.g., stocks belonging to the same sector show similar patterns. Capturing these types of relationships is essential to better understanding market dynamics and creating effective trading strategies accordingly. Although initially most of the approaches proposed in the literature treated each stock as isolated for prediction, a new line of work is actively exploring the joint prediction of multiple stocks. Most of these works integrate graph neural networks [25] to model such correlations in static [26, 27] or dynamic (i.e., learned directly from the model) [17] graphs.

5.3. Profit-Driven Optimization

Another aspect often overlooked in the design of ML/DL models in finance concerns the correct definition of the learning strategy according to the investment objective. Most of the approaches do not directly optimize the target of investment in terms of profit, even if they are interested in identifying the most profitable stocks. In other words, the stock selection task is typically formulated as a classification problem (to estimate the future trend of stocks) or a regression problem (to directly estimate the future price/return of stocks). However, correctly solving these tasks can lead to sub-optimal solutions in terms of profit [12, 4]. Consider the toy example shown in Table 1, where two regressors (R1 and R2) and two classifiers (C1, C2) are respectively used to predict the return and the trend of 5 stocks. As can be seen, the worst-performing models (i.e., R2 and C2) are able to select the most profitable top-1 stock compared to the best-performing methods (i.e., R1 and C1). Following this direction, a new line of work has suggested adopting a ranking approach, which is closer to the problem of selecting the most profitable stocks [27, 17]. Instead

1Note that in the regression the top-1 stock is selected based on the higher predicted return, while in the classification based on the higher probability of the positive trend.
of predicting, for example, the return of stocks (as in a regression task), the goal here is to sort the stocks by decreasing return. In this way the stocks that perform better than others will appear first in the ranking and will be selected by a topk-based trading strategy.

6. Evaluation

The goal of the evaluation step in the financial domain is twofold. First, the predictive ability of the ML/DL model must be evaluated and, second, the performance of the trading strategy must be analyzed. The latter is built on top of the model’s predictions and varies depending on the type of supervised task used for model training. In a classification scenario, the up and down predictions are interpreted as buy and sell signals. In the regression and ranking scenarios the (top-k) stocks with the highest predictions are bought and those (top-k) with the lowest predictions are sold.

To achieve this, different metrics and evaluation procedures for both tasks have been proposed. With regard to evaluation metrics, a distinction is made between model metrics and portfolio metrics, depending on whether they evaluate the model or the strategy. Commonly used portfolio metrics include return, Sharpe ratio, and Sortino ratio. Regarding the evaluation procedures, instead, an out-of-sample evaluation scheme is typically used to evaluate the effectiveness of the model (e.g., cross-validation is the most commonly adopted solution), while a backtesting technique is employed to analyze the performance of the trading strategy.

However, there are still several problems that practitioners may encounter during the evaluation process. They arise mainly from the tendency of models to overfit and the presence of serial correlation in the data.

6.1. Time/Serial correlations

Although most financial models are evaluated in standard cross-validation (i.e., an extension of out-of-sample evaluation to multiple train-test splits), it is not the ideal evaluation tool for financial data. This is due to the existence of various forms of temporal correlations in the data which create leakages, or implicit overlaps between train and test data, compromising the reliability of the evaluation process. To mitigate this issue, a new cross-validation scheme has been proposed in [8], where purging and embargoing techniques are applied to remove such dependencies. More specifically, the purging technique removes from the train set all observations whose labels overlapped in time with those included in the test set. In a task that predicts monthly stock returns, for example, this means creating a window of at least 30 days between train and test observations. On the other hand, embargoing creates a further gap between train and test sets when the latter precedes the train set in time. This is done to avoid that it contains information that is highly correlated with the next train set.

6.2. Overfitting

A very common condition in financial machine learning is overfitting, i.e., the poor ability to generalize to new data. This condition mainly affects backtesting strategies, although it is also common in financial model training [18]. Backtest overfitting occurs when a strategy is over-optimized on a specific backtest scheme, resulting in poor performance if the backtest is changed. Most of the trading strategies are affected by this condition, as they are evaluated exclusively with the popular walk-forward (WF) scheme. With this procedure, the historical data is divided into two sets, the in-sample and out-of-sample periods. The strategy is developed and optimized during the in-sample period and evaluated during the out-of-sample period. The scheme is repeated by moving the in-sample and out-of-sample periods forward in time.

Although this procedure has the advantage of providing a clear historical interpretation of the performance of a strategy, it has the disadvantage of testing a single scenario obtained by splitting the data only in the forward direction. To mitigate this problem, Combinatorial Purged Cross-Validation (CPCV) has recently been proposed [8]. It modifies a traditional K-Fold cross-validation scheme by generating all possible combinations of train-test splits having \( m > 1 \) folds as test set and the remaining folds for the train set, while purging train observations that contain leaked information. Unlike traditional cross-validation methods, the test sets are not used to compute performance metrics directly. Instead, they are divided into groups, each representing an independent evaluation path. In this way, multiple backtest paths are evaluated instead of a single one, reducing backtest overfitting.

7. Conclusions

In this paper, we have provided a systematic review of the main pitfalls affecting fintech practitioners in developing stock selection strategies, and we have collected the main solutions used to mediate them. Starting from the data preparation step, the most adopted practices are the use of clipping techniques to reduce abnormal patterns, the correct management of price-adjusted and fundamental data to avoid look-ahead bias, and the inclusion of delisted stocks to limit survivorship bias. In the featurization phase, the main solutions to manage the inhomogeneity and non-stationarity of stock series are the adoption of sampling techniques based on volume or dollar bars and the transformation of price series.
into yield series. The use of graph-based models and the modeling of the output in probabilistic terms (e.g., with quantile or gaussian losses) represent the most used techniques for capturing correlations between stocks and managing their chaotic nature. Finally, purging and embargoing combined with advanced cross-validation and backtesting procedures are the main practices employed to obtain bias-free evaluation strategies.

References


