Summary of Stock Selection Strategy and Backtesting System

Yuhuang Chen

May 1, 2024

1 Stock Selection Strategy

1.1 Brief Description

There are more than 5000 stocks in the Chinese stock market with different quality and features, our objective is to meticulously select a defined quantity of stocks to assemble our investment portfolio. To achieve the purpose, we would utilize data analysis to create a grading scheme of stocks which can be dated back to the empirical study by Fama and French.

In the realm of quantitative investment, the data is the most important part. What is the data we need. On every trading day, our analysis involves processing a two-dimensional dataframe, typically sized at around 5000 stocks by 300 factors. In this matrix, each row signifies an individual stock, while each column corresponds to a distinct stock factor. These factors encompass a wide range of data, from financial statement metrics such as earnings, cash flows, and debt ratios, to market-related information like price and volume.

Our grading mechanism operates akin to a stratified filtration system. At each layer, stocks are evaluated and scored, with only a subset progressing to the subsequent level. Scores are derived from a function of factors, illustrated as

$$S_1 = g(f_1, f_2, f_3) \tag{1}$$

If f_1 represents beta, f_2 represents value, f_3 represents size, this aligns with the traditional three-factor model.

Stocks are then ranked based on their scores, and, as an example, only the top half are retained for further consideration.

Progressing to the next stage involves a similar selection process, guided by a different scoring function, such as

$$S_2 = h(f_4, f_5, f_6) \tag{2}$$

This layered filtering process continues until the pool of stocks is condensed to a predetermined target number, for instance, 10. Ultimately, the strategy ends in the selection of the top 10 stocks to construct the portfolio.

An alternative method for analyzing the stock selection strategy is through the use of a decision tree model. In this model, each score is represented as a node within the tree. Each node effectively segments the stock universe based on the scores assigned. The portfolio ultimately selected corresponds to the leaf node at the terminus of the tree.

1.2 Frequently Asked Questions and Answers

Q1. *What is the frequency of the strategy?*

A1. Daily.

Q2. *What is the portfolio rebalancing process?*

A2. Each day, our strategy generates a revised list of stocks. To rebalance the portfolio, we compare this new list against our current holdings. Stocks present in our portfolio but absent from the day's list are liquidated, while those newly included in today's recommendations but not currently held are acquired.

Q3. *How is the position allocation conducted?*

A3. We employ an equal-weighted allocation approach for distributing positions. Specifically, each selected stock is allocated an identical proportion of capital.

Q4. *How do you evaluate the scores?*

A4. I follow the methodology from Turan Bali 's book *Empirical Asset Pricing*. For more information, I posted some blogs regarding this at this website https://quaizz.github.io/year-archive/

Q5. *How do you prevent over-fitting?*

A5. It is challenging to assert with complete certainty that a strategy will not overfit, as such assertions largely depend on future performance. Thus, the most reliable method to evaluate potential overfitting is through out-of-sample testing, which I have conducted over the past two years. Given that my strategies have consistently demonstrated superior performance in these two-year out-of-sample tests, I am confident they do not result from overfitting.

Looking back two years, there were several measures I employed to ensure the robustness of my strategies against overfitting. One fundamental approach was to simplify the model by reducing the number of parameters. For instance, when developing a scoring strategy based on multiple factors, rather than fine-tuning the weightings, I assigned equal weights. This method assumes that simplicity enhances robustness; no amount of fine-tuning can transform ineffective inputs into valuable outputs. Moreover, robust factors maintain their efficacy even without weighted differentiation.

Additionally, part of the validation process included stress testing to evaluate how the strategies performed during bear markets. For example, the 2018 downturn in the Chinese stock market was particularly telling. Some strategies that yielded high returns in average conditions faced substantial losses of up to 30% during this period, leading to their exclusion from further consideration.

Another effective method involved shifting the sample space. By designing a strategy that initially encompassed the entire market and then recalibrating it to focus on specific indices, such as the CSI300 or CSI500, I could assess whether the strategy still delivered high excess returns. Successful adaptation to these narrower indices would indicate that the strategy's logic holds universally across different market segments.

1.3 Further Analysis

From the description of the strategy, we can find that the key point that can differentiate the strategies is the scores. Fundamentally, the essence of crafting a successful strategy hinges on identifying effective scoring formulas that adeptly pinpoint stocks with high profit potential.

However, delving into a simplistic yet illustrative mathematical analogy, let's consider a fixed function g as referenced in equation (1). With a pool of 300 factors at our disposal, theoretically, we could generate upwards of 2^{300} unique formulations of g. Even if we just focus on scores composed of a specific number of factors, such as 7 or 8, the combination we need to search would be above 10^{10}. For example, $\binom{300}{6} \approx 9.6 \times 10^{11}$, $\binom{300}{7} \approx 4 \times 10^{13}$, $\binom{300}{8} \approx 1.48 \times 10^{15}$.

Tackling this immense variety through manual experimentation would be an unfeasible endeavor, far exceeding the bounds of a human lifetime. This predicament underscores the necessity for an automated process to streamline the evaluation.

This system automates the testing of different scoring algorithms against historical data, expediting the identification of effective strategies while ensuring a comprehensive and efficient exploration of potential configurations.

2 Alpha-generating Backtesting System

2.1 Brief Description

The purpose of the system is to find S that can generate sustainable profits. To achieve this, we embark on an exhaustive process of generating all variations of S through g. Each variant undergoes a thorough backtesting phase, where its performance and returns are recorded.

Following this extensive evaluation, those scoring functions demonstrating superior returns are shortlisted for further scrutiny. The next step involves a risk analysis of these high-performing scores to ascertain their viability for real-world investment applications. This approach ensures that only strategies with a balanced profile of high returns and manageable risks are considered for deployment in actual investment scenarios.

2.2 Backtest Implementations

The backtesting process is conducted in an event-driven framework, which accommodates the complexity and realism of financial markets.

On each simulated trading day, the system recalculates scores and adjusts the portfolio accordingly, incorporating considerations such as taxes, slippage, and commission fees to closely replicate actual trading conditions.

Two primary issues warrant our meticulous attention during this process.

The foremost concern involves liquidity challenges, particularly prevalent in the Chinese stock market due to regulatory price limits on stock movements. Prices of stocks are confined to fluctuate within a designated range — typically, between 90% and 110% of the previous trading day's closing price. Encountering the upper price limit poses a significant barrier, as purchases become impractical due to the scarcity of sellers. Consequently, if our strategy signals to buy a stock that has reached its upper limit, the attempt should realistically be deemed unsuccessful, mirroring the actual trading limitation where such transactions are unfeasible.

To deal with the issue, we enhance the dataframe by incorporating columns that specify the upper and lower price limits for each stock. Execution of orders within our backtesting system will proceed only if the stock's unadjusted opening price remains within these predefined bounds. Should the opening price breach these limits, indicating that the stock has hit either its upper or lower limit, the corresponding trade signal will be categorized as a failed execution. This refinement ensures our backtesting framework more accurately reflects the constraints encountered in real-world trading scenarios.

Another challenge encountered during backtesting pertains to stock splits and dividends. These events impact stock prices significantly, and addressing them within the backtesting framework can be time-consuming. To tackle this, we opt for an approximation method that balances speed and accuracy effectively.

For dividends, we employ adjusted prices in our return calculations. This approach ensures smooth continuity in the price series, thereby facilitating accurate performance analysis.

Stock splits present a more complex scenario. To elucidate, consider an example where an investor purchases 1,000 shares of stock BYD at a price of 100 on January 1. On January 15, the stock undergoes a 2-for-1 split, adjusting the share price to 52 and doubling the investor's shares to 2,000. If the investor sells all 2,000 shares at a price of 55 on January 20, the realized return is 10%. The return calculation in this case is represented by the formula:

$$1000 \times 100 \times \left(\frac{55}{50} - 1 \right) \tag{3}$$

which can be generalized as:

$$\text{number of shares purchased (before split)} \times \text{unadjusted purchase price} \times \left(\frac{\text{adjusted selling price}}{\text{adjusted purchase price}} - 1 \right) \tag{4}$$

This formula adeptly accommodates the adjustments necessitated by splits, ensuring the backtesting remains both swift and precise. It's important to note that in an ideal scenario, maintaining a precise track of the share count in our portfolio and adjusting it in response to split events would more closely mirror real-world conditions. However, this approach requires significant computational resources, here we circumvent this time-consuming process (we do not need to calculate the 2000 and update it when we backtest). This method allows us to manage stock splits effectively without compromising on the speed or the accuracy of the backtesting process.

2.3 Prune the Brute Force Search

Given the astronomical number of potential scoring functions g that require evaluation through backtesting, it's practical to acknowledge that many of these scores might prove ineffective, essentially not worth the time and resources needed for comprehensive backtesting. To streamline this process and bypass the evaluation of non-promising scores, a strategic approach can be employed to pre-screen candidate scores.

This approach involves assessing the similarity between a new candidate score and those that have already undergone backtesting. This preliminary step aims to filter out scores for backtesting in two scenarios: we proceed with scores that are distinct from previously tested ones or those that closely mirror the characteristics of scores that have previously shown promising results. Conversely, if a candidate score closely resembles one identified as ineffective or 'garbage', it is preemptively excluded from further testing. This methodology is grounded in the rationale that minor tweaks to fundamentally flawed scores are unlikely to transform their performance substantially.

When substantial amount of tested scores have been accumulated, approximately 30% of the candidates can be skipped. This approach not only conserves valuable computational resources and time but also enhances the overall efficiency of the search process by focusing efforts on exploring novel or proven effective scoring mechanisms.

2.4 Performance Optimization in Software Development

Several noteworthy techniques have been instrumental in optimizing our strategy development process, primarily contributed by my teammates. I'd like to highlight their significance:

To address the challenges posed by the extensive array of candidate scores, we've encountered memory limitations that could potentially lead to system overloads. The solution is to generate and evaluate candidate scores in smaller batches. This approach aligns with our memory capacity, preventing crashes and ensuring smooth operation.

Given the extended duration required for processing vast datasets, there's an increased risk of power outages, which can corrupt or destroy files that have already been processed. To mitigate this risk, a power outage protection mechanism is implemented, whereby backup copies of files are automatically saved at midnight each day. This precaution ensures that the integrity of our data is maintained, safeguarding the progress of our computational tasks against unexpected interruptions.

The integration of the Numba package marks a significant enhancement in our workflow. By enabling the pre-compilation of Python code, similar to C programming methodologies, Numba significantly boosts execution speeds. This is achieved by dynamically compiling Python functions into machine code at runtime, streamlining computations.

Furthermore, the adoption of multi-threading techniques has substantially augmented our strategy's efficiency. By utilizing a multi-core CPU setup, we can conduct parallel processing, markedly diminishing computation times and elevating overall productivity.

Some typical choice of pandas function can also help. For instance, employing the bitwise NOT operator $\sim$ for negation is more efficient than comparing boolean values with `== False`.

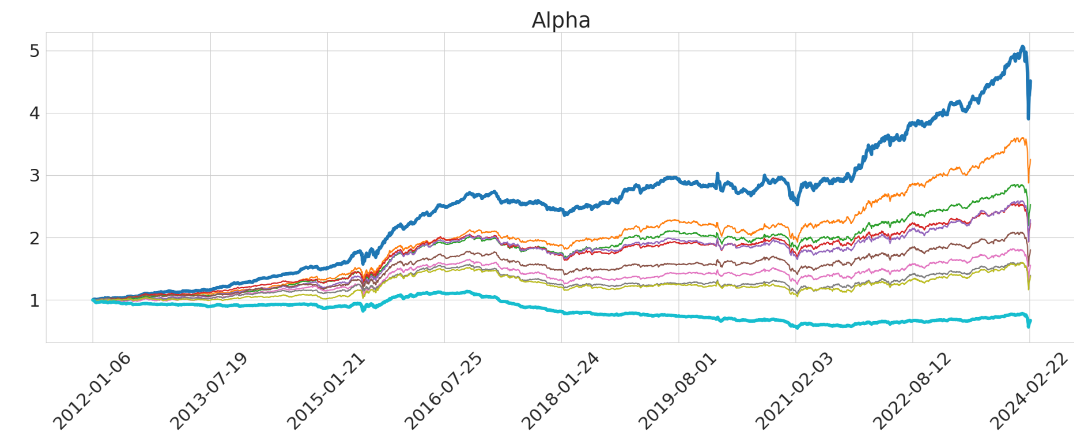
2.5 Risk Analysis of the Profitable Scores

Once we've identified factors capable of yielding satisfactory profits, it's essential to delve deeper into their analysis. This involves addressing several critical questions to ensure the robustness and reliability of our strategy. We need to determine whether the returns demonstrate a consistent trend over time. It's also crucial to assess the portfolio's resilience during significant market downturns — could we experience a drastic drop in returns? Furthermore, understanding the risk attribution associated with our scoring system is vital. Our goal is to verify the sustainability of the scores we generate.

To facilitate this in-depth examination, we can leverage visualization techniques. Graphical representations will allow us to closely examine the behavior of these factors under various market conditions, providing valuable insights into their performance stability and risk characteristics. This approach will enable us to make informed decisions and refine our investment strategy accordingly.

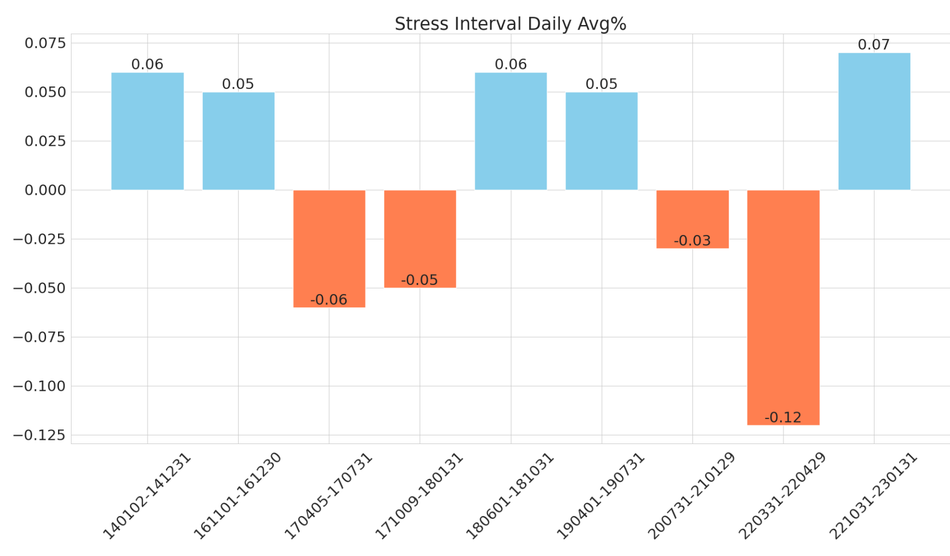
A number of insightful visualizations can assist in this deeper analysis, with one of the key techniques being group analysis. In this approach, we segment the entire market into 10 distinct groups, rebalancing these groups on a monthly basis to observe their respective excess returns. By examining whether the returns are stratified across these groups — meaning that the returns systematically vary from one group to another based on their assigned scores — we can infer the effectiveness of our factor. If a clear stratification is observed, where groups with higher-rated contracts consistently outperform those with lower-rated ones, it would indicate that our factor is indeed effective in predicting relative performance within the market.

Cumulative Excessive Return:



Conducting a stress test on the scoring system is another valuable analysis method. This involves evaluating the performance of our scores during historical periods of market stress. By analyzing how well the scoring system holds up under various challenging conditions — including market downturns, economic crises — we can assess its robustness. If the scores consistently demonstrate resilience and continue to perform well across multiple stress periods, it indicates a level of robustness that merits further exploration.

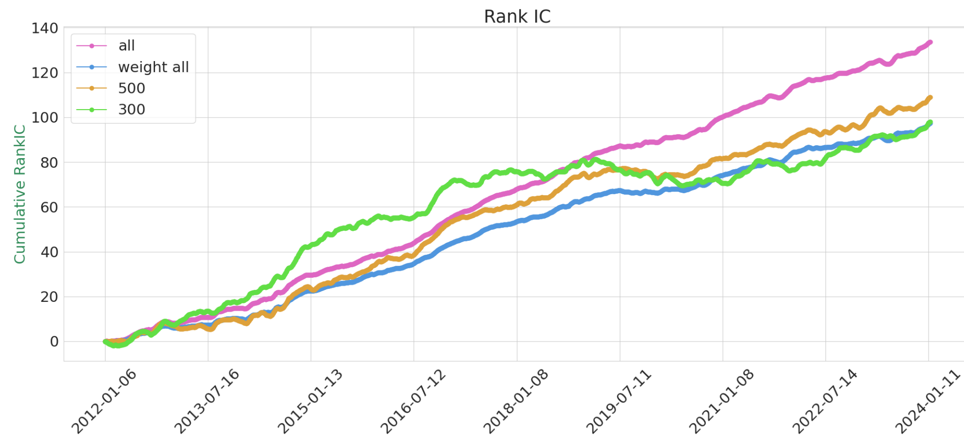
Daily Return Under Stress Test:



Another insightful method to evaluate the efficacy of our scoring system is through the calculation and analysis of the Rank Information Coefficient (Rank IC). On a monthly basis, we calculate the Rank IC for our scores, accumulating these values over time. By plotting the cumulative sum of these Rank IC values, we can visually assess the trend of this metric.

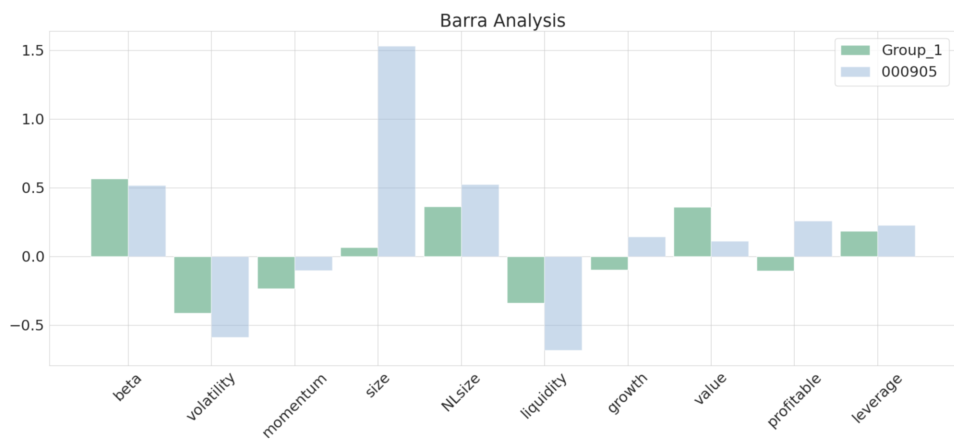
If the plotted curve exhibits a consistent upward trajectory, it suggests that our scoring system possesses the ability to perform effectively across various scenarios. A stable or increasing cumulative Rank IC indicates that there's a positive and persistent correlation between the scores and subsequent returns, affirming the utility of the scores in different market conditions.

Cumulative Rank IC:



For assessing risk attribution, we can implement a Barra risk analysis on the portfolio. This approach enables us to examine the portfolio's risk exposures to various common factors in the market.

Barra:



3 Backtesting System of Commodities Market: A Working Project

3.1 Brief Description

After the end of the internship where we focused on stocks, we tried to extrapolate the idea of the backtesting system into the commodities market in China, just as an unsupervised individual project for fun. The basic idea is that, we can replace the dataframe of stocks with the dataframe of future contracts. A similar grading scheme can be applied and we can long the contracts with the highest grade and short the contracts with the lowest grades.

The most fundamental difference between the two systems is the difference of trading rules between the two markets and others are basically, the same.

3.2 Backtesting Implementations

Several distinction between the two markets would deserve our special attention.

In the commodities market, each commodity is represented by multiple futures contracts, each with its own settlement date. This contrasts with the stock market, where a single company typically corresponds to a single stock. To navigate this complexity in commodities, investors often focus on the "dominant contract," which is identified by the highest trading volume and open interest. This approach helps in selecting the most liquid contract for investment purposes. Additionally, the data provider creates an index for each commodity based on the price of the dominant contract. This results in a continuous time series, which seamlessly incorporates changes in the dominant contract. When there is a shift to a new dominant contract, the price from the new contract is merged with the previous prices, ensuring the continuity of the time series.

The other alternation is that in commodities market the unit of contract we long or short can be influenced by the risk preference. Here we adopt the formula from Andreas Clenow's book *Following the Trend*, which is

$$\text{number of contracts} = \frac{\text{value of portfolio} \times \text{risk coefficient}}{ATR_{20} \times \text{multiplier of contract}} \tag{5}$$

Here the risk coefficient is a hyper parameter that reflects the our risk preference, typically it can be set to a number between 0.005 to 0.01.

ATR_{20} is the 20 day average of True Range(TR), which can be calculated as

$$TR_t = \max\{High_t, Close_{t-1}\} - \min\{Low_t, Close_{t-1}\} \tag{6}$$

Here the $High$, Low, $Close$, are the corresponding price of the commodity index.

Liquidity remains a crucial consideration. Some futures contracts may exhibit low trading activity, making it challenging to execute trades based on certain signals successfully. To mitigate this issue, it's advisable to focus on commodities demonstrating high liquidity. Specifically, commodities indexes with an average daily trading volume exceeding 10,000 over the past year are considered. This criterion aims to minimize the risk of insufficient market depth at the time of executing trades.

However, liquidity concerns persist. To address this, trades are scheduled for execution at a fixed time of 9:10 a.m. In our trading simulation, we verify the presence of historical trading activity at this specific minute. A trade is deemed executable only if there was positive trading volume at that time. Additionally, when planning a trade, we compare the intended trading volume with the maximum trading volume recorded for the contract on that day. The executed trade volume is then determined by the lesser of these two values, ensuring that the trade does not exceed available market liquidity. This approach helps in creating a more realistic simulation of trading conditions, enhancing the reliability of our trading strategy in environments with variable liquidity.

Another critical aspect of trading in the futures market is the finite life span of contracts. Futures contracts have set expiration dates, necessitating investors to transition or "roll" their investment from a contract nearing its expiration to one with a later expiration date.

In the process of rebalancing our portfolio, attention must be paid not only to adjusting the allocation across different commodities based on our investment strategy but also to managing the intricacies of contract rollovers.

There are occasions when the rollover process, based on the dominant contract strategy, might not proceed as planned, leading to situations where we retain certain contracts until their last trading day. To circumvent the complications of physical settlement, it becomes imperative to liquidate all such contracts in our possession on their final trading day.

Our simulated trading operates on a daily basis. Each day, we execute the following procedures:

- **Filter out the dominant contract with liquidity:** We identify and select the dominant contract for each commodity, focusing on those with significant liquidity.

- **Process the grading scheme:** Similar to stock trading, we apply a grading scheme to each contract. Trading signals are generated to go long on the contract with the highest score and short on the contract with the lowest score.

- **Rebalance the portfolio according to the signal:** The portfolio is adjusted in response to the signals, aligning our holdings with the current trading strategy.

- **Liquidate contracts approaching the last trading date:** Contracts nearing their last trading day are liquidated to circumvent the complications of physical settlement, ensuring timely closure of all positions.

3.3 Next Things To Do

For now we are still studying the trading execution part to make it fit the online backtesting system. Now we still do not have enough factors to go through.