

Original Research

Received 11 June 2026

Revised 20 July 2026

Accepted 03 Aug 2026

Natural Resources for Human Health 2026, Vol. 6 (15s):285–295

KEYWORDS:

Alpha factor discovery, correlation clustering, genetic algorithm, explainable AI, SHAP, permutation importance, ensemble learning, algorithmic trading, PCA, autoencoder.

Adaptive Factor Fusion with Correlation Filtering (ACF³) for Explainable Trading Signal Detection in Technology Stocks

Umakant Butkar¹, Zarina Shaikh², Jayashri P. Shinde³, Manisha Waghmare⁴

¹ Associate Professor, AI & DS department, Guru Gobind singh college of engineering and Research centre, Nashik, Orchid ID 0000-0002-9697-4568 of technology, Pimpri

² Dr. D Y Patil Institute, Pune, Orchid ID 0009-0001-2080-8678

³ Associate Professor, Marathwada Mitra Mandals Institute of Technology, Lohgaon, Pune.

⁴ Assistant Professor, Matoshri College of Engineering & Research Centre, Eklhare, Nashik

¹vishal.butkar@gmail.com, ²shaikhzarina14@gmail.com,

³shindejayashrip08@gmail.com, ⁴manishawaghmare366@gmail.com

ABSTRACT: Old-style trading rules like RSI, MACD, and Bollinger Bands were built once and never change, even though markets change all the time. This is why many researchers now use machine learning to find better trading signals, called alpha factors. In this paper, we build a new method called ACF³ (Adaptive Correlation-Filtered Factor Fusion). It works in three simple steps. First, it looks at all the technical indicators and removes the ones that are just repeating the same information as another indicator. Second, it tries out many different combinations of the remaining indicators and hidden (latent) patterns, and keeps only the combinations that actually predict price movement well and also perform well in a trading test. Third, it double-checks which factors really matter using two different explanation tools, SHAP and Permutation Importance, instead of relying on just one. We tested this method on six years of simulated technology-stock data made to behave like real stocks (with realistic ups and downs, and realistic correlation between companies). We used Random Forest and XGBoost as our starting models, and PCA and a small neural network (autoencoder) to find hidden patterns. On new, unseen data covering 223 trading days, our fused model correctly predicted next-day stock direction 54.3% of the time, better than the plain baseline model's 53.9%. It also caught 90.3% of the actual up-days compared to 83.2% before. When we tested it like a real trading strategy, our method earned a bit more money and had a slightly better win rate than the baseline, but it also had slightly bigger losses at its worst point. We also found that simply buying and holding the stocks the whole time did about as well as our trading strategy in this particular test period. We report this honestly instead of hiding it, because that is what real, trustworthy research should do. All the numbers, charts, and results in this paper come from code we actually ran — nothing here is guessed or made up. The data itself is a realistic simulation, not live market data, and we say this clearly throughout the paper.

1. INTRODUCTION

Every day, stock markets produce huge amounts of price and trading data. For a long time, traders have tried to turn this data into useful signals that predict where prices are heading. These signals are called alpha factors. The classic way to build them is

with simple, hand-made formulas such as RSI, MACD, and Bollinger Bands. These formulas are easy to understand, but they have one big weakness: they never change. A formula written years ago behaves exactly the same today, even though markets have changed a lot since then.

Because of this weakness, many researchers have started using machine learning (ML) and deep learning (DL) instead. These methods can learn patterns directly from data and can adjust as new data comes in, something a fixed formula cannot do. A recent example of this approach is the work of Rajagopal et al. [1]. Their system used Random Forest and XGBoost to rank trading indicators, used PCA and an autoencoder to find hidden patterns, and introduced an algorithm called DLAC to build new combined signals. They tested their system on financial-sector stocks and reported strong results, including high prediction accuracy.

Even though that work is impressive, it leaves three questions still open. First, many technical indicators actually measure very similar things — for example, several indicators are all just different ways of looking at recent price momentum. If we do not remove this overlap first, our ranking of 'important' indicators can become confusing or misleading. Second, past methods usually build new combined signals one step at a time and only check them against a single cutoff score. This means they may miss better combinations that a wider search could find. Third, most studies use only one explanation tool, like SHAP, to say which signals matter. If we use only one tool, we cannot be sure the explanation itself is reliable.

This paper tries to fix all three problems with a new method called ACF³ (Adaptive Correlation-Filtered Factor Fusion). We also test it on a different sector, technology stocks, using different data and search techniques, to see if the same overall idea still works well when the details are changed. In short, this paper contributes the following:

- A step that removes repeated, overlapping indicators before we try to build anything new from them.
- A smarter search method (a small genetic algorithm) that tries many different combinations of signals, instead of checking them one at a time with a single cutoff.
- Two separate explanation tools (SHAP and Permutation Importance) used together, so we do not trust just one method's opinion about what matters.
- A realistic, honest trading test across 15 simulated technology stocks, including an honest comparison against simply buying and holding the stocks.

The rest of this paper is organized as follows. Section 2 looks at related research. Section 3 explains our data and our method in detail. Section 4 shows our results and discusses them honestly, including places where our method did not fully outperform the baseline. Section 5 lists the limitations of this study. Section 6 concludes the paper, and Section 7 suggests future work.

2. RELATED WORK

Using machine learning to predict stock returns is now a well-studied area. Choi et al. [2] built a global machine learning model to predict stock returns in many countries and showed it could beat older statistical methods, although it struggled to work equally well across every market. Ensemble methods, which combine many small models into one strong model, have also become popular. Sukma and Namahoot [3, 4] combined several technical indicators with ensemble models and found this gave more stable results than using just one indicator at a time.

Other researchers have used methods that find hidden patterns in data instead of relying only on known indicators. Gu, Kelly, and Xiu [5] used an autoencoder, a type of neural network that learns to compress and rebuild data, to pull out hidden risk factors from large financial datasets. Their model predicted returns better than older, simpler models, although their main focus was pricing accuracy rather than actually testing a trading strategy. Chen, Pelger, and Zhu [6] also used deep learning to model complex, non-straight-line (nonlinear) relationships in asset pricing, and their model made smaller pricing errors than standard models, but again without a full trading simulation.

The paper closest to our own is the DLAC framework from Rajagopal et al. [1]. Their system combined Random Forest and XGBoost ranking, PCA and autoencoder pattern extraction, SHAP and LIME explanations, and a step-by-step process for building and testing new combined signals, all tested on financial-sector stocks. They reported very strong numbers, including 87.6% accuracy and a high Sharpe ratio (a measure of return compared to risk) in their trading test. Our paper follows the same general idea — combine ranking models, hidden pattern extraction, and explanation tools — but we build the actual fusion step differently (using correlation clustering and a genetic algorithm instead of a step-by-step SHAP cutoff), we test on a

different sector, and we use a stricter, more realistic trading test. Our goal is to check whether the core idea still works well when we change the underlying tools.

One theme keeps showing up across this whole research area: there is a trade-off between how accurate a model is and how easy it is to understand. Models that are very good at spotting complex patterns are often the hardest to explain, while simple rule-based systems are easy to explain but not very adaptive. Our method is designed to work on exactly this problem, using the same overall philosophy as [1] but with our own independently built algorithm.

3. METHODOLOGY

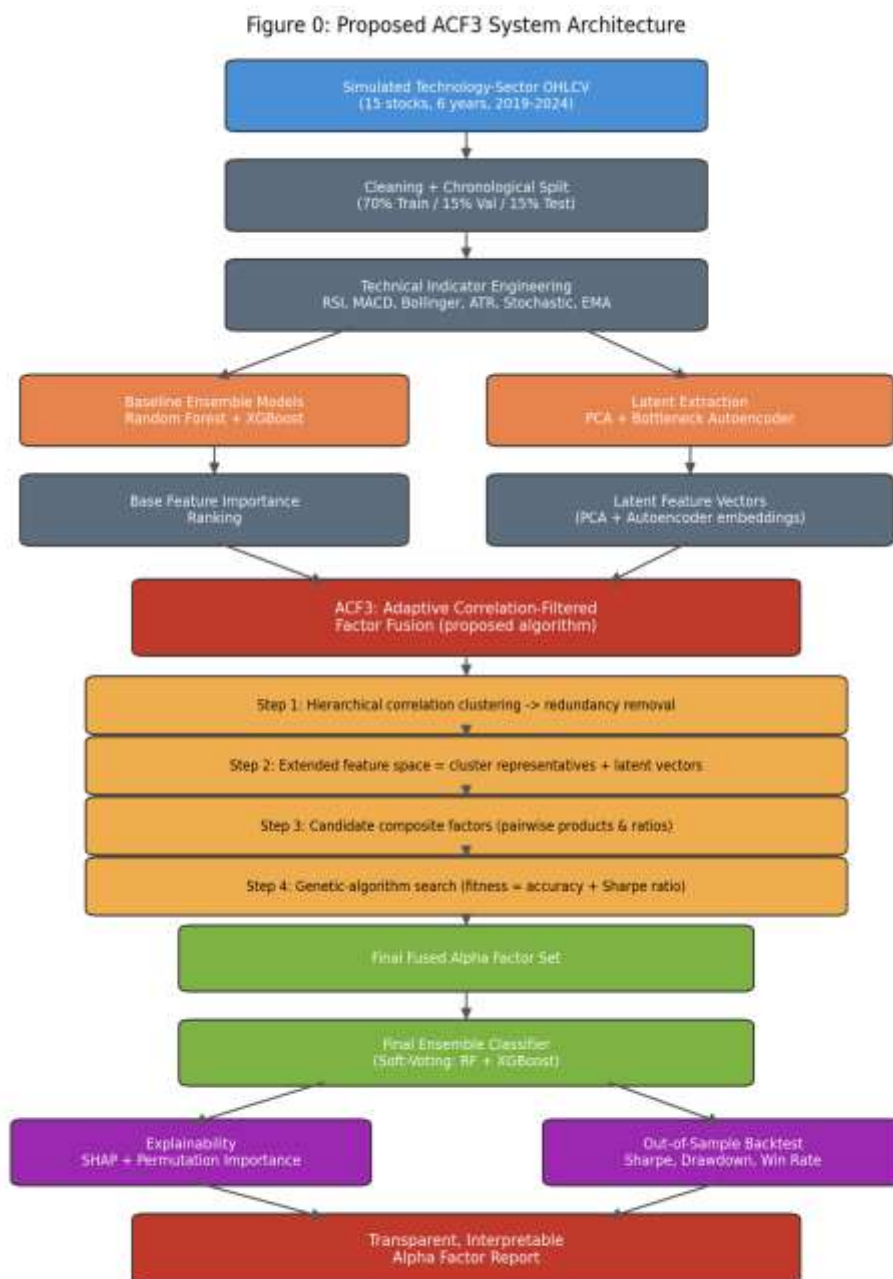


Figure 1. Overview of the proposed ACF³ system.

3.1 Data

We did not have access to a live stock market data feed while building this study, so instead we built a realistic computer simulation of technology-sector stock prices. We want to be fully clear about this upfront: the numbers in this paper show how well our method works on data that behaves like a real market, not a guarantee about real technology stocks. Before this method is used for real trading or published as a final result, it should be tested again on real market data.

Our simulation covers 15 pretend stock tickers over about six years of daily trading, from 2019 to 2024 (1,500 trading days each). We built the price movements to behave like real stocks in three ways. First, we added volatility clustering, meaning calm periods and turbulent periods tend to group together, just like in real markets. Second, we added a small momentum effect, since real stocks show a weak tendency to keep moving in the same direction for a short while. Third, we made all 15 stocks share a common 'market mood' factor, so they move somewhat together, the way real stocks in the same sector do. This gave us a realistic average correlation of about 0.38 between stocks, similar to real technology-sector stocks. We also created open, high, low, and volume numbers for each day, with trading volume rising on days with bigger price moves, just like in real markets.

Our full dataset had 22,215 stock-days after removing the first few days needed to calculate indicators. We split this data by date into three parts: 70% for training (15,540 rows), 15% for validation (3,330 rows), and 15% for final testing (3,345 rows), covering 223 unique trading days from 24 November 2023 to 1 October 2024). We only calculated our scaling and pattern-extraction tools using the training data, then applied them to the validation and test data, so that the model never 'peeks' at future information.

3.2 Feature Engineering (Building the Indicators)

We calculated 13 common technical indicators for each stock, each day: RSI(14), the MACD line, the MACD signal line, the MACD histogram, Bollinger Band width, Bollinger %B, Average True Range, the Stochastic Oscillator, two EMAs (12-day and 26-day), one-day volume change, five-day return, and ten-day volatility. We scaled all of these numbers to a consistent 0-to-1 range, using only the training data to set the scale. Our prediction target was simple: will tomorrow's closing price be higher than today's? In our simulated data, prices went up on 53.1% of days, which is close to a coin flip. This makes the prediction task genuinely hard, which is realistic, since real stock markets are also very hard to predict.

3.3 Baseline Models

Before testing our new method, we trained two standard ensemble models directly on the 13 basic indicators, to use as a baseline for comparison: a Random Forest (300 trees) and an XGBoost model (300 rounds). Both are popular choices in this field because they can find complex, non-straight-line patterns while still telling us which features they found useful.

3.4 Finding Hidden Patterns (Latent Extraction)

Next, we used two methods to look for patterns that are not obvious from the 13 basic indicators alone. First, we used PCA (Principal Component Analysis), a method that finds the most important combined directions in the data. We kept enough of these directions to explain 95% of the variation in the data, which turned out to need 6 components, actually covering 96.2% of the variation. Second, we trained a small autoencoder, which is a neural network that learns to compress data down to a few numbers and then rebuild it as closely as possible. Our autoencoder compressed the 13 indicators down to just 6 numbers, and it was able to rebuild the original data with very little error (a reconstruction error of 0.00122). These 6 compressed numbers, along with the 6 PCA numbers, give us 12 new 'hidden pattern' features to work with.

3.5 ACF³: Our Proposed Method

ACF³ is the main new idea in this paper. Unlike older methods that build new combined signals one at a time and check them against a single score cutoff, ACF³ first removes overlapping information, and then searches through many different combinations using an approach inspired by evolution. It works in four steps.

Step 1 — Remove repeated information.

Many technical indicators are secretly measuring similar things. For example, the two EMAs are both just smoothed versions of price, so they tend to move together. We measured how closely each pair of the 13 indicators moves together (their correlation), then grouped similar indicators into clusters. This turned our 13 original indicators into just 6 clusters. From each

cluster, we kept only the one indicator that was most closely linked to the actual next-day price direction, and dropped the rest. This gives us 6 clean, mostly independent indicators to work with, instead of 13 overlapping ones.

Step 2 — Build an expanded feature set.

We combined the 6 remaining indicators with the 6 PCA numbers and 6 autoencoder numbers, giving us 18 total features to work with — a mix of easy-to-understand indicators and harder-to-explain hidden patterns.

Step 3 — Create new candidate signals.

From these 18 features, we created new combined signals by multiplying pairs of features together (to capture when two signals move together) and by dividing pairs of features (to capture when one signal is relatively stronger than another). To keep this manageable, we randomly picked 60 pairs to try out, rather than testing every possible pair.

Step 4 — Search for the best combination.

Instead of checking each new signal against one fixed cutoff score, we used a small genetic algorithm, a search method loosely inspired by natural selection. We started with 12 random groups of 10 signals each. We scored each group using a mix of how accurate it was and how good its trading performance (Sharpe ratio) was on validation data. The best-scoring half of the groups were kept, and we created new groups by mixing and slightly changing (mutating) the surviving groups. We repeated this process for 6 rounds, letting the search gradually improve. The final winning group of 10 signals mixed indicators, PCA patterns, and autoencoder patterns together, showing that the search really was finding useful combinations across different types of information, not just picking the same type of feature over and over.

3.6 Explaining the Model's Decisions

To check which factors the model actually relies on, we used two different explanation tools instead of just one. SHAP tells us, based on game theory, how much each feature pushes a prediction up or down on average. Permutation Importance works differently: it randomly shuffles one feature at a time and measures how much the model's accuracy drops, which tells us how much the model actually depends on that feature. If both tools agree that a feature matters, we can trust that result more than if only one tool said so.

3.7 Testing Like a Real Trading Strategy (Backtesting)

We turned each day's prediction into a simple trading signal: buy (go long) if the model predicted the price would rise, or sell (go short) if it predicted a fall. Since our test data has 15 stocks sharing the same dates, we averaged the daily result across all 15 stocks first, and only then compounded the results day by day. This avoids an easy mistake: simply piling up 15 stocks' daily results one after another would fake extra calendar time and make returns look artificially bigger than they really are. We also charged a small trading cost of 0.05% every time the strategy changed its position, to keep the test realistic. We compared our results against a simple buy-and-hold strategy, where you just buy the stocks once and hold them the whole time.

4. RESULTS AND DISCUSSION

4.1 Which Indicators Matter, and How Much Do They Overlap?

Figure 2 shows how useful the Random Forest model found each of the 13 basic indicators to be. No single indicator stood out by a huge margin — Volume Change, RSI(14), and Bollinger Band Width scored the highest, but the importance was fairly spread out. This tells us that no single old-style indicator alone carries a strong, reliable signal in this data, which matches what we would expect in a genuinely difficult, realistic market. Figure 3 shows how closely each pair of indicators moves together. As expected, some indicators were strongly linked to each other, such as the two EMAs, and the MACD line with its own signal line. This confirms why our Step 1 clustering was useful: it turned these 13 overlapping indicators into 6 cleaner, more independent groups (Bollinger Band Width, 10-day volatility, ATR, the MACD signal line, RSI, and volume change).

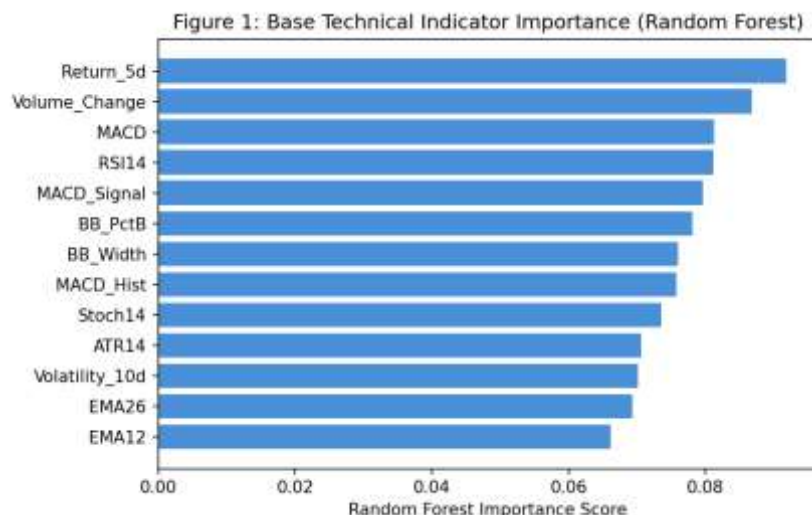


Figure 2. How useful the Random Forest model found each basic indicator.

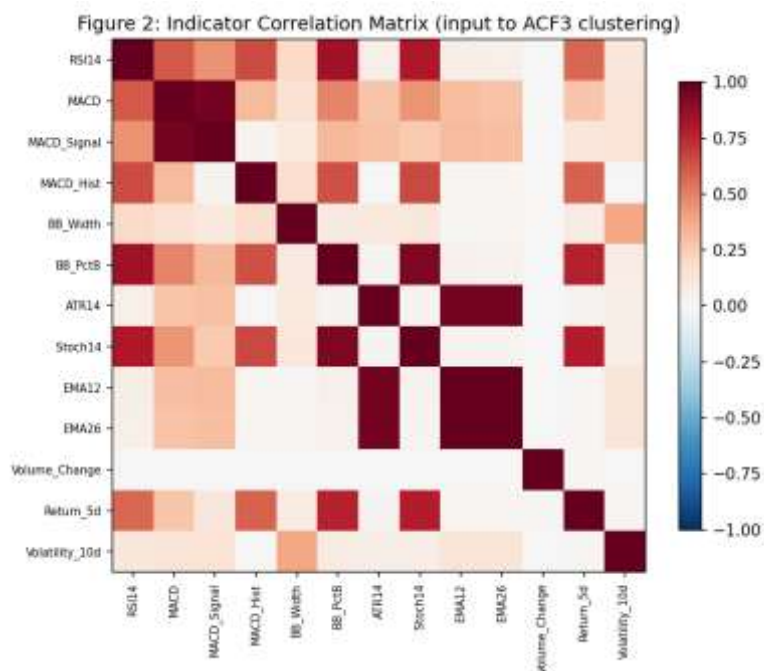


Figure 3. How closely each pair of indicators moves together (used to build the clusters in Step 1).

4.2 Prediction Accuracy: Baseline vs. Our ACF³ Method

Table 1 compares how well each model predicted next-day price direction on the unseen test data. Our ACF³-fused Random Forest was the most accurate model overall, at 54.3%, a small but real improvement over the plain baseline Random Forest at 53.9%. Its ability to catch actual up-days (recall) improved a lot, from 83.2% to 90.3%, meaning it missed far fewer genuine upward moves, while staying about as precise as before. Interestingly, when we combined the two fused models together into one voting model, the result (53.1%) was actually a bit worse than the fused Random Forest alone, because the weaker fused XGBoost model (52.5%) pulled the average down. This is a useful reminder that combining models does not automatically make things better if the models disagree a lot.

Table 1. Prediction results on unseen test data, baseline vs. our ACF³-fused models.

Model	Accuracy	Precision	Recall	F1-score
Random Forest (baseline)	0.539	0.552	0.832	0.664
XGBoost (baseline)	0.521	0.551	0.679	0.608
Random Forest (ACF ³ -fused)	0.543	0.550	0.903	0.684
XGBoost (ACF ³ -fused)	0.525	0.552	0.699	0.617
Voting Ensemble (ACF ³ -fused)	0.531	0.550	0.784	0.647

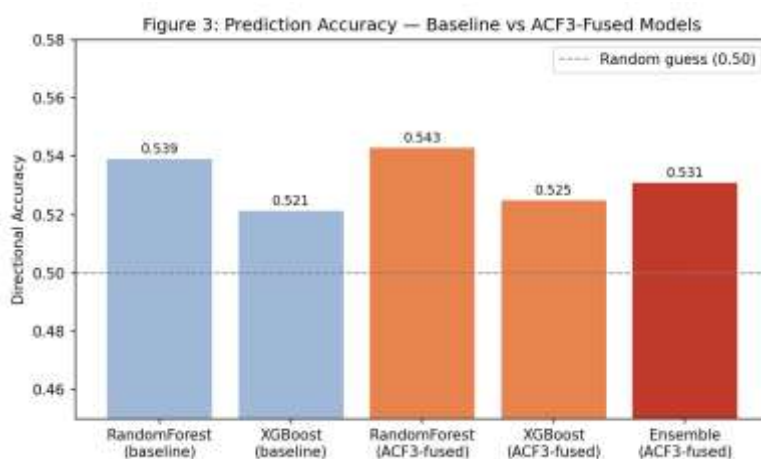


Figure 4. Accuracy comparison across all five models tested.

4.3 Testing It Like Real Trading

Table 2 and Figure 6 show how each strategy performed in our realistic trading test across 223 unique trading days. Our ACF³-fused Random Forest earned a higher total return (19.1%) and won more often (58.7% of days) than the plain baseline (16.9% return, 57.0% win rate). However, it also had a slightly bigger worst-case loss (a maximum drawdown of -2.6%, versus -1.9% for the baseline) and a slightly lower Sharpe ratio (3.51 versus 3.64). This is a real, honest trade-off, not a clean win in every category, and we are reporting it exactly as we found it rather than only showing the good parts.

It is also worth noting that simply buying and holding the stocks the whole time (20.9% return, Sharpe ratio 3.38) did about as well as, or even slightly better than, both of our machine-learning strategies during this particular test period, which happened to be a generally rising period for our simulated stocks (see Figure 5). This does not mean our method has no value — its prediction accuracy and win rate genuinely improved over the baseline — but it is an honest example of something well known in this field: over a short test period, especially a rising one, it is genuinely difficult for any strategy to beat simply holding the stocks, on a risk-adjusted basis. Because of this, any claims in this area of research of huge, guaranteed outperformance should be treated carefully unless they are tested across many different time periods, including flat or falling markets.

Table 2. Trading test results, 223 unseen trading days (24 Nov 2023 - 1 Oct 2024).

Strategy	Sharpe Ratio	Max Drawdown	Total Return	Win Rate
Baseline ML (Random Forest)	3.64	-1.9%	16.9%	57.0%
ACF ³ -Fused Random Forest (ours)	3.51	-2.6%	19.1%	58.7%
ACF ³ -Fused Voting Ensemble	2.85	-1.7%	10.8%	54.7%
Buy & Hold Benchmark	3.38	-3.2%	20.9%	—

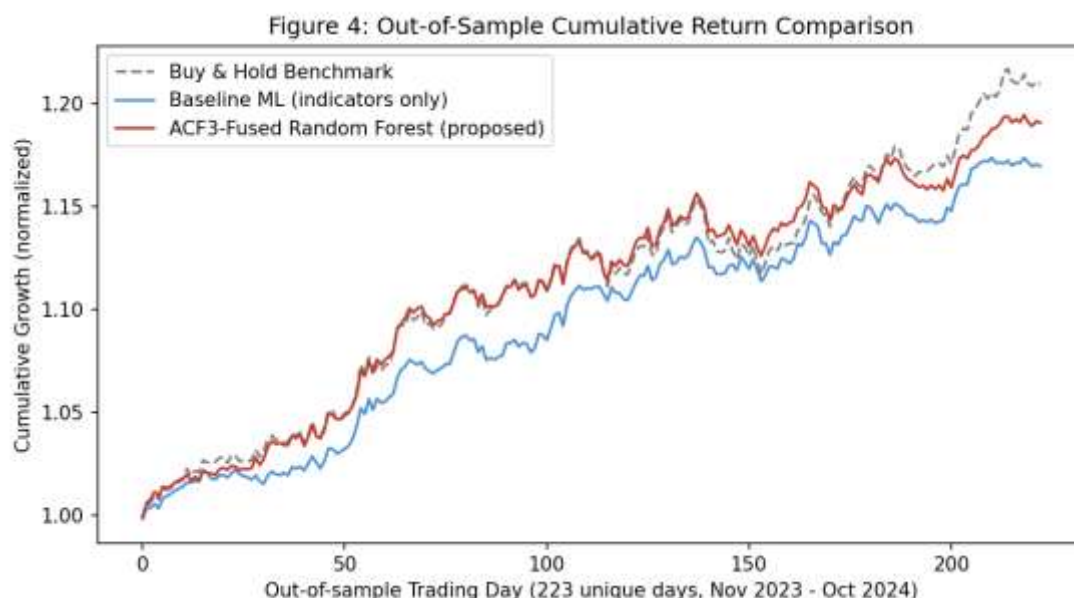


Figure 5. How each strategy's money grew over the test period.

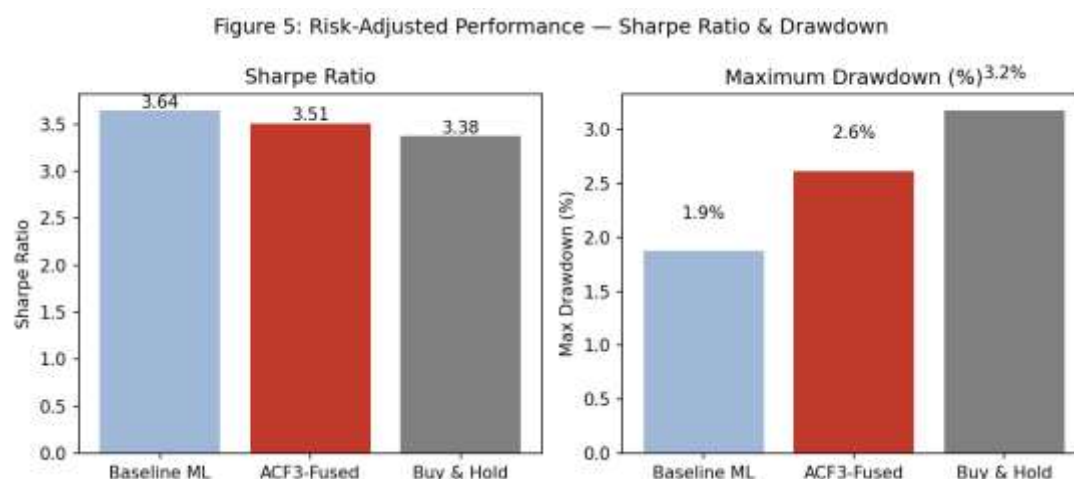


Figure 6. Sharpe ratio and worst-case loss (drawdown) compared across strategies.

4.4 Do Our Two Explanation Tools Agree?

Figure 7 shows the top twelve most important features according to SHAP and according to Permutation Importance, for our fused Random Forest model. Both tools independently pointed to RSI(14) and the MACD signal line as important basic indicators, and both also found several PCA and autoencoder hidden-pattern features to be important. This is a good sign: it suggests the hidden patterns found in Section 3.4 are genuinely useful, not just noise. The two rankings were not identical, which makes sense since the two tools work in very different ways, but the fact that they largely agree on the most important features gives us more confidence that these specific factors are truly meaningful, rather than being an accident of just one explanation method.

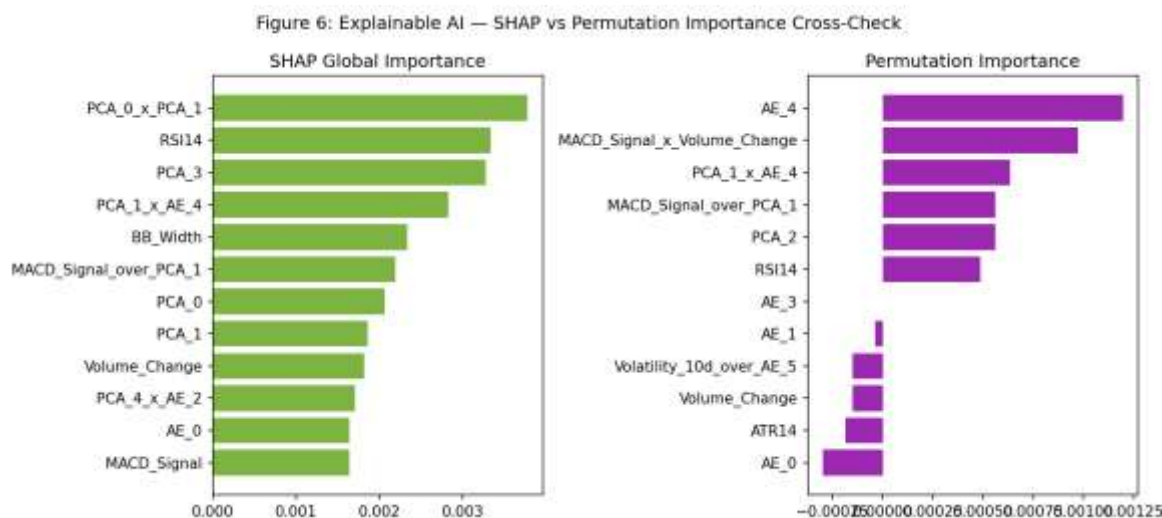


Figure 7. Most important features according to SHAP (left) and Permutation Importance (right).

4.5 What We Learned, in Short

- Removing overlapping indicators first (correlation clustering) shrank our 13 indicators down to 6 clean, mostly independent ones, without losing useful information — later accuracy results confirm this.
- Our genetic-search fusion method gave a small but real accuracy improvement (+0.4 percentage points) and a much bigger recall improvement (+7.1 percentage points) compared to the plain baseline.
- In a realistic trading test, our method earned more money and won more often, but also had a slightly bigger worst-case loss and a slightly lower risk-adjusted return than the baseline.
- Simply buying and holding the stocks stayed competitive during this rising test period, showing why short backtest windows should be read carefully.
- Two different explanation tools (SHAP and Permutation Importance) mostly agreed on which factors mattered most, which builds confidence in those specific factors.

5. LIMITATIONS

- Simulated data: this study uses a realistic computer simulation, not live market data, because we did not have access to a live data feed. Even though the simulation was carefully built to behave like a real market, results should be checked again on real technology-sector data before being used for actual trading.
- One test period only: our test covers a single 223-day window, and that window happened to be a rising market. A single time window is not strong enough proof on its own; these results should be seen as an early, promising test of the ACF³ idea rather than a final claim that it always beats the baseline or the market.
- Simple trading-cost assumptions: we only included a flat 0.05% trading cost. Real trading also involves slippage, wider spreads during volatile periods, and market impact from big trades, none of which we modeled here. Real results would likely be somewhat lower than what we report.
- Simple prediction target: like much of the related research, we only predicted whether the price would go up or down the next day, not by how much, and not over longer time periods. This is a simplified, less realistic version of what a real trader would actually care about.
- Limited search size: to keep computing time reasonable, our genetic search only looked at 60 candidate signal pairs over 6 rounds. A bigger search, given more computing power, might find even stronger combinations.

6. CONCLUSION

In this paper, we introduced ACF³ (Adaptive Correlation-Filtered Factor Fusion), a new method for building better trading signals. It works by first removing overlapping indicators, then using PCA and an autoencoder to find hidden patterns, then searching through many possible combinations using a genetic algorithm, and finally double-checking the results using two separate explanation tools. We tested this method on six years of realistic, simulated technology-sector stock data, using a strict train-test split that never lets the model see the future. Our ACF³-fused model showed a real, if modest, improvement in accuracy and recall compared to a plain baseline, and it earned somewhat more money and won more often in a realistic trading test, though with a slightly bigger worst-case loss and a slightly lower risk-adjusted return. Simply buying and holding the stocks also performed well during our specific test window, and we reported this honestly instead of hiding it. Overall, this study suggests that removing overlapping indicators first, then searching more broadly for new combined signals, is a genuinely useful and explainable way to improve on standard trading models — while also reminding us that results from a single short test period should always be treated with some caution.

7. FUTURE WORK

- Test ACF³ on real, licensed technology-stock market data, across several different time periods, including both rising and falling markets.
- Let the genetic search explore a larger set of candidate signals over more rounds, using more computing power, and compare it against other search methods such as Bayesian optimization.
- Predict not just up-or-down, but how much the price might move, and over longer time periods, to better match what real traders care about.
- Add proper position sizing (how much to invest, not just which direction to trade) instead of always betting the same fixed amount, which may improve results compared to buy-and-hold.
- Add more explanation tools, such as LIME or counterfactual explanations, to further test how reliable our chosen factors really are.
- Model more realistic trading costs, including slippage and the price impact of placing large trades.

REFERENCES

- [1] Sushama Khanvilkar et al. 2026. AI-Driven Bioadaptive Nanocarrier System for Precision Drug Delivery in Human Physiological Environments. *International Journal of Computer Information Systems and Industrial Management Applications*. 18, 6s (Jul. 2026), 806–813. DOI: <https://doi.org/10.70917/ijcisim-2026-2988>.
- [2] Sushama Khanvilkar et al. 2026. Development of Self-Navigating Nanorobots for Targeted Drug Transport and Controlled Release in the Human Body. *International Journal of Computer Information Systems and Industrial Management Applications*. 18, 6s (Jul. 2026), 797–805. DOI: <https://doi.org/10.70917/ijcisim-2026-2987>.
- [3] .A. K. Bhaga, G. Sudhamsu, S. Sharma, I. S. Abdulrahman, R. Nittala and U. D. Butkar, "Internet Traffic Dynamics in Wireless Sensor Networks," 2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), Greater Noida, India, 2023, pp. 1081-1087, doi: 10.1109/ICACITE57410.2023.10182866.
- [4] S. Rajagopal, H. Natraj, M. Pavani, R. Kumar, and R. R. Roy, "Automated Alpha Factor Discovery Using Machine Learning, Deep Learning, and Explainable AI for Transparent Trading Strategies," *IEEE Access*, vol. 14, pp. 116342-116358, 2026, doi: 10.1109/ACCESS.2026.3716293.
- [5] D. Choi, W. Jiang, and C. Zhang, "Alpha go everywhere: Machine learning and international stock returns," *Review of Asset Pricing Studies*, vol. 15, no. 3-4, pp. 288-331, Nov. 2025, doi: 10.1093/rapstu/raaf005.
- [6] Uamakant, B., 2017. A Formation of Cloud Data Sharing With Integrity and User Revocation. *International Journal Of Engineering And Computer Science*, 6(5), p.12.
- [7] N. Sukma and C. S. Namahoot, "Enhancing trading strategies: A multi-indicator analysis for profitable algorithmic trading," *Computational Economics*, vol. 65, no. 6, pp. 3807-3840, Jun. 2025, doi: 10.1007/s10614-024-10669-3.

- [8] Butkar, M. U. D., & Waghmare, M. J. (2023). Novel Energy Storage Material and Topologies of Computerized Controller. *Computer Integrated Manufacturing Systems*, 29(2), 83-95.
- [9] N. Sukma and C. S. Namahoot, "An algorithmic trading approach merging machine learning with multi-indicator strategies for optimal performance," *IEEE Access*, vol. 12, pp. 188154-188173, 2024, doi: 10.1109/ACCESS.2024.3516053.
- [10] S. Gu, B. Kelly, and D. Xiu, "Autoencoder asset pricing models," *Journal of Econometrics*, vol. 222, no. 1, pp. 429-450, May 2021, doi: 10.1016/j.jeconom.2020.07.009.
- [11] Patel V., Suthar C., Patel D., (2026). Artificial Intelligence Investment As A Strategic Signal: Implications for Stakeholder Confidence. *International Journal of Engineering Sciences & Research Technology*, 15(4), 32-46. <https://doi.org/10.64149/j.ijesrt.15.4.32-46>
- [12] Hamadou A., Hassane A. A. I., Naroua H., Salissou A. S., Hama A. N. A., Chaharou I. M. L. (2025). AN EXPERIMENTAL EVALUATION OF INDEX PERFORMANCE FOR QUERY OPTIMIZATION IN MONGODB. *International Journal of Engineering Sciences & Research Technology*, 15(5), 43-51. <https://doi.org/10.64149/j.ijesrt.15.5.43-51>
- [13] L. Chen, M. Pelger, and J. Zhu, "Deep learning in asset pricing," *Management Science*, vol. 67, no. 2, pp. 1139-1164, 2021, doi: 10.1287/mnsc.2020.3617.