

# Advanced Programming 2025

## Machine Learning for Weekly ETF Allocation

A Time-Series Backtesting and Walk-Forward Study

Final Project Report

Thomas Remandet

[thomas.remandet@unil.ch](mailto:thomas.remandet@unil.ch)

Student ID: 21422506

December 30, 2025

### Abstract

This project investigates whether simple machine learning (ML) classifiers can extract predictive signals from historical prices to support a weekly allocation strategy across five liquid ETFs (SPY, QQQ, EEM, TLT, GLD). Daily prices are resampled to weekly closes and converted to weekly returns. A feature set combining lagged returns, momentum and volatility indicators, and macro/regime proxies (risk-on/off, correlations, breadth) is constructed without look-ahead. For each ETF, three models (Logistic Regression, Random Forest, XGBoost) are trained using a chronological 80/20 split and evaluated with accuracy, precision, recall, and ROC-AUC. The best model per ETF is then used to build a probability-based allocation rule and compared against financial benchmarks (equal-weight, buy-and-hold SPY, momentum, and rolling Markowitz optimizations). Robustness is assessed via temporal cross-validation and walk-forward analysis (2-year train, 6-month test, rolling windows). Results show weak and often statistically insignificant classification skill (ROC-AUC typically near 0.5), while portfolio outcomes are largely driven by benchmark effects and allocation rules rather than genuine predictive power. The main contribution is an end-to-end, reproducible pipeline for time-series ML evaluation and systematic comparison against strong baseline strategies.

**Keywords:** time series, ETFs, machine learning, backtesting, walk-forward validation, portfolio allocation

## Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	Background and Motivation . . . . .	4
1.2	Problem Statement . . . . .	4
1.3	Objectives . . . . .	4
1.4	Report Organization . . . . .	4
<b>2</b>	<b>Literature Review / Related Work</b>	<b>4</b>
2.1	Momentum Strategies and Behavioral Finance . . . . .	4
2.2	Portfolio Optimization . . . . .	5
2.3	Machine Learning in Asset Pricing . . . . .	5
<b>3</b>	<b>Methodology</b>	<b>5</b>
3.1	Data Description . . . . .	5
3.1.1	Source and Content . . . . .	5
3.1.2	Data Source and Reliability . . . . .	6
3.1.3	Preprocessing and Missing Data . . . . .	6
3.1.4	Targets . . . . .	6
3.2	Approach . . . . .	6
3.2.1	Feature Engineering . . . . .	6
3.2.2	Models . . . . .	7
3.2.3	Evaluation Protocol . . . . .	7
3.2.4	Trading Strategies (Economic Evaluation) . . . . .	7
3.3	Implementation . . . . .	8
3.4	Interactive Dashboard . . . . .	8
3.4.1	Example Feature/Target Construction . . . . .	8
<b>4</b>	<b>Results</b>	<b>8</b>
4.1	Experimental Setup . . . . .	8
4.2	Performance Evaluation . . . . .	9
4.2.1	Predictive Performance (Classification) . . . . .	9
4.2.2	Economic Performance (Portfolio Backtesting) . . . . .	10
4.3	Portfolio Evolution and Risk Analysis . . . . .	11
4.4	Prediction Quality Analysis . . . . .	12
4.5	Robustness: Walk-Forward Results . . . . .	13
<b>5</b>	<b>Discussion</b>	<b>13</b>
5.1	Interpretation . . . . .	13
5.2	Methodological Corrections During Development . . . . .	13
5.3	Key Limitations . . . . .	14
5.4	What Worked Well . . . . .	15
<b>6</b>	<b>Conclusion and Future Work</b>	<b>15</b>
6.1	Answering the Research Question . . . . .	15
6.2	Summary . . . . .	15
6.3	Future Improvements . . . . .	16
<b>A</b>	<b>References</b>	<b>17</b>
<b>B</b>	<b>AI Tools Used</b>	<b>18</b>
<b>B</b>	<b>Reproducibility Notes</b>	<b>18</b>

<b>C Additional Data Tables</b>	<b>18</b>
C.1 Complete ML Model Performance Results . . . . .	18
C.2 Walk-Forward Analysis Summary . . . . .	19
C.3 Statistical Significance Tests . . . . .	19

## 1 Introduction

### 1.1 Background and Motivation

Forecasting short-horizon asset returns is widely considered difficult due to low signal-to-noise ratios and market efficiency effects [1]. Nonetheless, practitioners often attempt to exploit weak predictive signals using systematic methods, combining technical indicators, regime features, and robust backtesting. This project focuses on a controlled academic setting where we accept that results may be inconclusive, prioritizing methodological correctness and reproducibility over ambitious claims.

The idea for this project originated from discussions with two financial professionals whose perspectives shaped the project's direction. First, an independent wealth manager (*Gestionnaire de Fortune Independant*, GFI) whose investment approach relied primarily on diversified ETF portfolios raised the question of whether simple predictive models could provide incremental value in a systematic allocation framework. Second, Mr. Gyger, Director of Investment Policy at Banque Cantonale Vaudoise (BCV), provided valuable insights on momentum-based strategies and their theoretical foundations in behavioral finance. These exchanges highlighted the growing interest in machine learning techniques for asset allocation while emphasizing the importance of understanding underlying market dynamics rooted in investor behavior.

### 1.2 Problem Statement

Given weekly historical returns for five broad ETFs (SPY, QQQ, EEM, TLT, GLD), can supervised ML classifiers predict the sign of next-week returns better than chance? If so, can these predictions improve a weekly allocation strategy compared to strong baselines such as momentum and Markowitz mean-variance optimization?

### 1.3 Objectives

This project aims to build a clean time-series pipeline without data leakage for weekly return prediction, comparing three ML models per ETF: Logistic Regression, Random Forest, and XGBoost. We evaluate both predictive performance through classification metrics and economic performance through backtested portfolio metrics, while validating robustness through temporal cross-validation and walk-forward testing across multiple market regimes.

### 1.4 Report Organization

The remainder of this report is organized as follows. Section 2 reviews related work in momentum strategies, portfolio optimization, and machine learning for financial prediction. Section 3 details the data sources, feature engineering approach, model selection, and evaluation protocol. Section 4 reports the empirical results from both classification and portfolio perspectives. Section 5 interprets the findings, discusses methodological corrections made during development, and acknowledges key limitations. Finally, Section 6 concludes with answers to the research question and proposes directions for future work.

## 2 Literature Review / Related Work

### 2.1 Momentum Strategies and Behavioral Finance

Momentum strategies rank assets by recent performance and allocate to winners, exploiting the well-documented momentum effect in equities and across asset classes [2]. Carhart [7] formalized momentum as a risk factor in asset pricing models, while Asness et al. [8] demonstrated that momentum effects persist across diverse markets and asset classes globally.

The theoretical foundation for momentum lies in behavioral finance, where investor psychology creates systematic patterns in asset prices. Kahneman and Tversky's [3] prospect theory demonstrates how cognitive biases affect decision-making under uncertainty, while Thaler's [5] work integrates these psychological insights into economic and financial frameworks.

A key behavioral mechanism driving momentum is the herding effect: investors are attracted to rising assets and tend to sell declining ones, creating self-reinforcing trends. However, momentum strategies face inherent risks, particularly the phenomenon of mean reversion when trends become overextended. When the deviation between an asset's price and its long-term average (such as a 200-day moving average) becomes too large, the probability of reversal increases significantly. This tension between trend-following and mean-reversion represents a fundamental challenge in momentum-based allocation, requiring careful calibration of entry and exit signals.

## 2.2 Portfolio Optimization

Portfolio optimization methods based on mean-variance trade-offs originate from Markowitz [4], and rolling implementations are frequently used as strong benchmarks when transaction costs are ignored. These approaches explicitly balance expected returns against portfolio risk, though they require estimation of covariance matrices that can be unstable with limited data. DeMiguel et al. [11] show that the naive equal-weight ( $1/N$ ) portfolio often outperforms optimized portfolios out-of-sample due to estimation error, highlighting the practical challenges of portfolio optimization with limited data.

## 2.3 Machine Learning in Asset Pricing

On the ML side, recent research emphasizes that predictive signals in financial returns are weak and require careful validation through techniques such as walk-forward testing, avoiding look-ahead bias, and comparing against strong benchmarks. Large-scale ML studies in asset pricing show improvements mainly through richer features and large cross-sections, but also highlight instability across time periods [6]. In small-universe settings with only five ETFs, overfitting and "model selection on the test set" are common pitfalls that can lead to misleadingly optimistic results.

De Prado [9] emphasizes the critical importance of avoiding common methodological errors in financial machine learning, including look-ahead bias, data snooping, and inadequate validation procedures. Bailey et al. [10] demonstrate how backtest overfitting can produce seemingly impressive results that fail to generalize out-of-sample, a concern particularly relevant when testing multiple strategies or model configurations.

This project is positioned as a rigorous baseline study with modest feature engineering, transparent models, strong benchmark comparisons, and explicit robustness checks. Rather than claiming superior performance, we focus on establishing a methodologically sound framework for time-series ML evaluation in portfolio allocation.

## 3 Methodology

### 3.1 Data Description

#### 3.1.1 Source and Content

The dataset contains daily prices for five ETFs representing major asset classes: SPY (US large-cap equities), QQQ (US technology stocks), EEM (emerging markets equities), TLT (long-term US Treasury bonds), and GLD (gold). Data are resampled to weekly frequency using Friday close prices. The effective weekly period after feature construction spans from February 26, 2016 to November 21, 2025, yielding 509 weekly observations with 44 engineered features.

### 3.1.2 Data Source and Reliability

The initial version of the project relied on a financial market API to retrieve historical ETF prices. However, for the purpose of producing a stable, reproducible, and submission-ready academic project, the final dataset was constructed using historical ETF price series obtained from the CEDIF (*Centre de Documentation en Finance*) [16] via Internef. This choice ensures consistent historical coverage, eliminates API availability constraints, and allows full reproducibility of results without external dependencies.

### 3.1.3 Preprocessing and Missing Data

Daily prices are read from a CSV file, converted to numeric values, and indexed by date before being resampled to weekly closes. Weekly returns are computed as the percentage change in prices:  $r_t = P_t/P_{t-1} - 1$ , where  $P_t$  represents the closing price at time  $t$ .

Missing values arise naturally from rolling window calculations (8-week, 12-week, and 26-week windows), affecting approximately the first 26 weeks of data. Rather than using forward-fill or interpolation methods that could introduce look-ahead bias, incomplete observations are removed via `dropna()`, reducing the dataset from approximately 520 to 509 usable weeks (a 2% loss). This conservative approach prioritizes temporal integrity over sample size.

Outlier events, including extreme market moves during the COVID-19 crisis (SPY declined 12% in a single week in March 2020), are deliberately preserved without winsorization or removal. These extreme observations represent genuine market conditions that any robust trading strategy must handle, and their exclusion would create an artificially smooth dataset that does not reflect real-world deployment challenges.

### 3.1.4 Targets

For each ETF, a binary target is created as  $y_t = \mathbb{1}[r_{t+1} > 0]$ , indicating whether the next week's return will be positive. Critically, the label at time  $t$  refers to next week's return direction, ensuring temporal integrity since features at  $t$  only use information available up to  $t - 1$ . This strict temporal alignment prevents look-ahead bias, a common pitfall in financial machine learning applications [9].

## 3.2 Approach

### 3.2.1 Feature Engineering

The feature set combines two categories of predictors designed to capture both asset-specific dynamics and broader market regimes.

**ETF-specific technical features** include lagged weekly returns for the past 1 to 4 weeks, rolling momentum calculated over 4-week and 12-week windows, rolling volatility computed over 8-week windows, Relative Strength Index (RSI) over 14 periods, moving average ratios comparing current price to 20-week moving averages, and volume ratios relative to 20-week average volume. All features are properly shifted to avoid information leakage.

**Macro and regime features** capture broader market dynamics through a risk-on/off proxy (SPY minus TLT momentum), a flight-to-safety indicator (GLD minus SPY momentum), rolling correlations between major indices (SPY–QQQ, SPY–EEM), market breadth measured as the percentage of ETFs with positive 4-week returns, and cross-sectional return dispersion calculated as the standard deviation of returns across all five ETFs.

This comprehensive approach yields 44 total features after preprocessing, consisting of 35 technical indicators and 9 macro/regime proxies. The feature set deliberately focuses on price-based and volume-based signals, reflecting the momentum and mean-reversion dynamics dis-

cussed in the behavioral finance literature, while avoiding fundamental or macroeconomic data that would require additional data sources.

### 3.2.2 Models

For each ETF, three classifiers are trained to represent different modeling approaches. Logistic Regression serves as a linear baseline, Random Forest [13] provides a nonlinear ensemble method capable of capturing interactions, and XGBoost [14] implements gradient boosting for potentially superior performance on structured data.

Class imbalance in weekly return direction is relatively mild, with positive week proportions ranging from 47.3% (TLT) to 55.4% (QQQ). Nonetheless, we apply class weighting (`class_weight='balanced'` for Logistic Regression and Random Forest, `scale_pos_weight` for XGBoost) to prevent models from defaulting to majority-class predictions and to encourage learning of discriminative patterns.

Hyperparameters are fixed a priori without extensive tuning: Random Forest uses 50 trees with maximum depth 5, while XGBoost employs 50 trees with depth 3 and learning rate 0.1. While this approach avoids overfitting the test set through hyperparameter search, it represents a limitation that future work could address through nested cross-validation.

### 3.2.3 Evaluation Protocol

The evaluation protocol consists of three layers designed to assess both predictive quality and robustness across market conditions.

The **primary split** follows a chronological 80/20 division, training on the first 407 weeks (approximately 8 years from 2016-2023) and testing on the final 102 weeks (approximately 2 years from 2023-2025). This temporal ordering respects causality and simulates realistic deployment where models trained on historical data must perform on future unseen periods.

We assess model quality through standard **classification metrics** including accuracy, precision, recall, and ROC-AUC. Additionally, we conduct binomial tests to determine whether observed accuracy levels are statistically distinguishable from random guessing (50% null hypothesis).

To validate **robustness across regimes**, we employ walk-forward analysis using 15 rolling windows, each training on 104 weeks (2 years) and testing on the subsequent 26 weeks (6 months) with non-overlapping test periods. This approach reveals whether performance is stable across different market conditions or highly regime-dependent.

### 3.2.4 Trading Strategies (Economic Evaluation)

Predicted probabilities are transformed into portfolio weights through two ML-based strategies and compared against multiple established benchmarks.

The **base ML Strategy** ranks the five ETFs by predicted probability of positive returns and allocates capital equally among the top three (33.33% each), rebalancing weekly. This ranking-based approach proves more robust than absolute probability thresholds, as it ensures consistent diversification even when all predicted probabilities fall below 0.50 (which occurs in approximately 13% of weeks). Portfolios are long-only with full investment at all times.

The **ML Strategy with Risk Management** extends the base approach by incorporating three additional controls: a 3% stop-loss rule to exit positions experiencing large weekly losses, a trailing stop mechanism that reduces exposure by 50% when drawdown from peak exceeds 10%, and volatility scaling that adjusts position sizes to target 15% annualized volatility based on 8-week rolling estimates.

These ML strategies are compared against five benchmarks: simple **buy-and-hold SPY**, **equal-weight** allocation across all five ETFs with weekly rebalancing, **momentum (4-week)**

strategy that ranks ETFs by recent 4-week returns and allocates equally to the top three, and two **rolling Markowitz optimizations** targeting either minimum variance or maximum Sharpe ratio [12] using 104-week rolling windows. The Markowitz strategies represent optimistic upper bounds as they ignore transaction costs, allow short positions, and assume stable covariance matrices.

### 3.3 Implementation

The project is implemented in Python using standard data science libraries including `pandas` for data manipulation, `numpy` for numerical operations, `scikit-learn` [15] for machine learning models, `xgboost` for gradient boosting, and `matplotlib` for visualization. Code is organized by functional responsibility: `src/data_loader.py` handles data loading and weekly resampling, `src/features.py` manages feature and target creation with explicit temporal integrity checks, `src/models.py` implements model training and evaluation, while `src/strategy.py` and `src/benchmarks.py` execute strategy backtests. Finally, `src/validation.py` performs temporal cross-validation and walk-forward analysis.

A `verify_temporal_integrity()` function automatically validates that all features use only past information and that targets refer to future returns, providing an additional safeguard against look-ahead bias.

### 3.4 Interactive Dashboard

In addition to the static analysis presented in this report, an interactive dashboard was developed using Streamlit for the web interface and Plotly for interactive visualizations. The dashboard enables dynamic comparison of strategies, inspection of cumulative returns and drawdowns, exploration of walk-forward results across market regimes, and detailed examination of prediction quality by ETF. AI-assisted tools were used to accelerate layout design and visualization refinement. While not required for the core evaluation, this component enhances interpretability and provides an intuitive way to communicate results beyond static figures.

#### 3.4.1 Example Feature/Target Construction

The following code snippet illustrates the leakage-safe construction of targets and features:

```
1 # returns: weekly return series r_t
2 target = (returns.shift(-1) > 0).astype(int)    # y_t = 1[r_{t+1} > 0]
3 features[f"{etf}_lag_1w"] = returns.shift(1)    # uses only past info
```

Listing 1: Leakage-safe target construction (next-week direction)

## 4 Results

### 4.1 Experimental Setup

The final dataset contains 509 weekly observations after feature alignment and removal of incomplete records. Using an 80/20 chronological split yields 407 training weeks and 102 test weeks. As an academic simplification, the analysis ignores transaction costs, slippage, and borrowing constraints, which would reduce real-world performance. Estimated transaction costs of 5-10 basis points per trade would reduce the ML Strategy's annualized return by approximately 3-4%, likely making it underperform simple buy-and-hold after frictions.

## 4.2 Performance Evaluation

### 4.2.1 Predictive Performance (Classification)

Table 1 reports out-of-sample metrics on the held-out test set for the best-performing model per ETF. The selected models are Random Forest for SPY, EEM, and GLD, and XGBoost for QQQ and TLT, chosen based on maximum ROC-AUC on a validation subset.

Table 1: Out-of-sample ML metrics for the selected best model per ETF (80/20 split)

ETF	Best Model	Accuracy	Precision	Recall	ROC-AUC
SPY	Random Forest	0.500	0.579	0.550	0.506
QQQ	XGBoost	0.608	0.617	0.847	0.639
EEM	Random Forest	0.539	0.600	0.610	0.521
TLT	XGBoost	0.520	0.526	0.577	0.512
GLD	Random Forest	0.559	0.730	0.435	0.571

Despite occasional improvements, notably for QQQ with an accuracy of 60.8% and ROC-AUC of 0.639, most scores remain close to 0.5, suggesting limited predictive power. Binomial significance tests reveal that only QQQ achieves statistically significant accuracy above the 50% random baseline ( $p = 0.018$  at the 5% level), while the other four ETFs fail to reject the null hypothesis of random guessing. The high precision for GLD (0.730) comes at the cost of low recall (0.435), indicating very conservative predictions that correctly identify positive weeks when they occur but miss many opportunities.

Figure 1 shows a detailed comparison of model performance across ETFs and metrics.

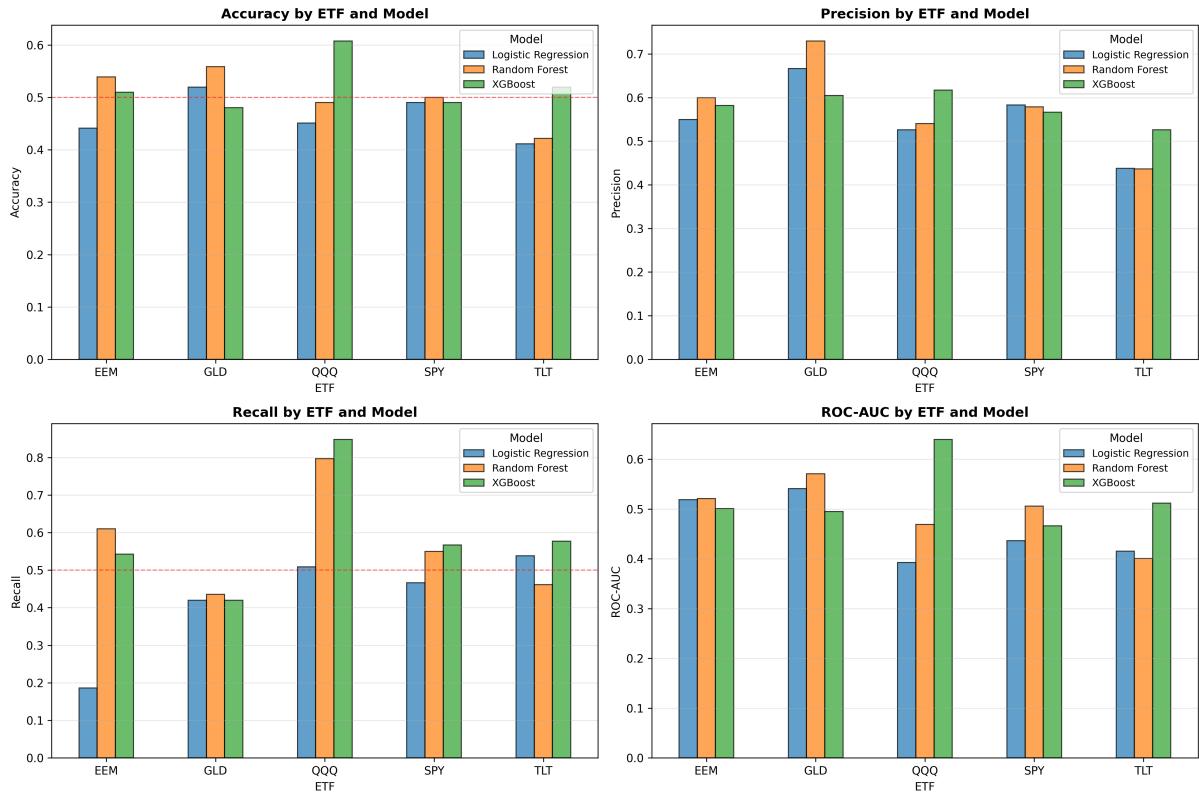


Figure 1: ML model performance comparison across ETFs

QQQ's superior performance may reflect characteristics specific to technology stocks during

the test period, including higher volatility (28% vs 18% for SPY) and stronger momentum effects that are more amenable to short-term prediction. Bonds (TLT) and commodities (GLD) exhibit more mean-reverting behavior that challenges momentum-based features.

#### 4.2.2 Economic Performance (Portfolio Backtesting)

Table 2 summarizes the backtest performance across all strategies on the test period. The results show that traditional Markowitz strategies and momentum approaches outperform ML-based portfolios under the assumed frictionless conditions.

Table 2: Backtest performance on the test period (weekly rebalancing; costs ignored)

Strategy	Total	Annual	Vol	Sharpe	Max DD	Win
Markowitz (Min Var)	0.469	0.217	0.091	2.372	-0.052	0.578
Markowitz (Sharpe)	0.615	0.277	0.123	2.259	-0.071	0.618
Momentum (4W)	0.516	0.229	0.117	1.962	-0.055	0.619
Equal Weight	0.432	0.201	0.105	1.919	-0.080	0.598
ML Strategy	0.411	0.194	0.131	1.479	-0.113	0.604
Buy & Hold SPY	0.432	0.201	0.149	1.346	-0.171	0.598
ML + Risk Mgmt	0.320	0.154	0.121	1.274	-0.110	0.604

The Markowitz minimum variance strategy achieves the highest Sharpe ratio (2.372) with the lowest maximum drawdown (-5.2%), demonstrating the value of explicit risk management through portfolio optimization. The momentum strategy also performs well with a Sharpe ratio of 1.962 and a win rate of 61.9%, validating the momentum anomaly documented in the literature [2]. In contrast, the base ML Strategy achieves a Sharpe ratio of 1.479, ranking fifth among seven strategies. Notably, the ML Strategy with additional risk management paradoxically performs worse (Sharpe of 1.274), as the return drag from protective rules outweighs the modest reduction in volatility.

Figure 2 shows the comprehensive backtesting results including cumulative performance, Sharpe ratios, risk-return profiles, and win rates.

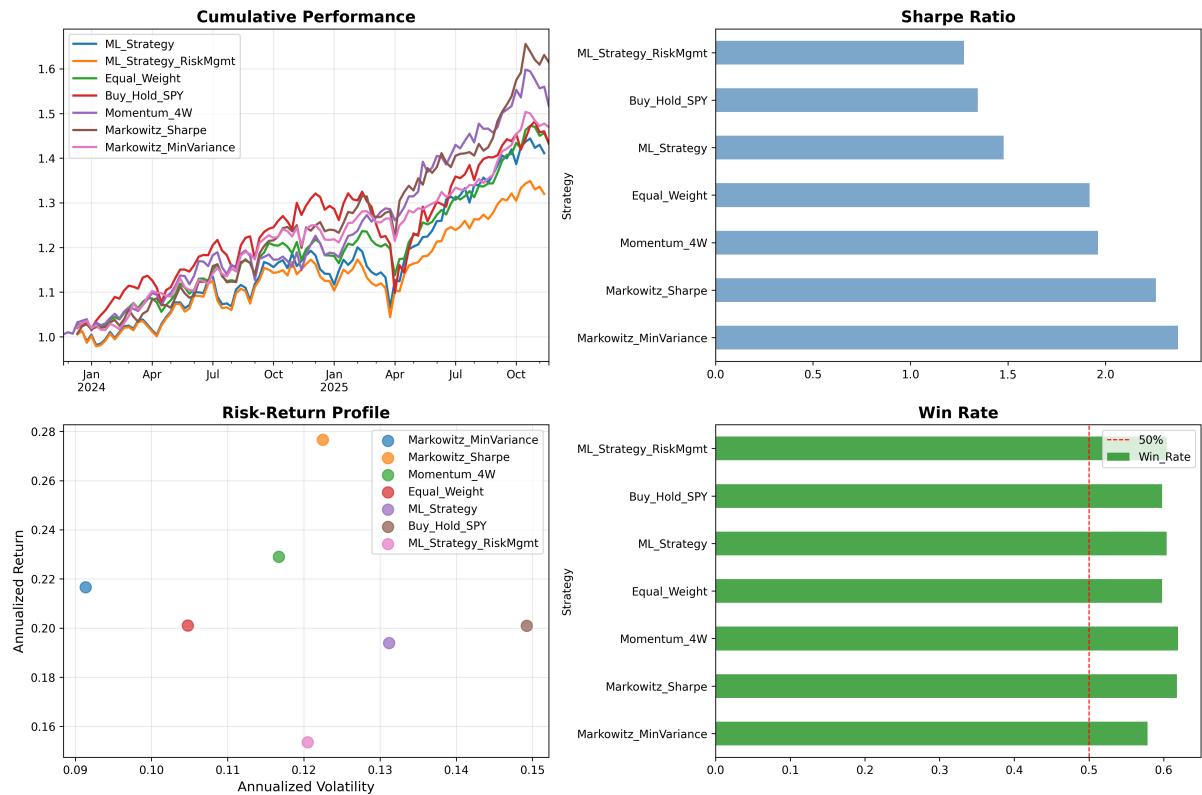


Figure 2: Comprehensive backtesting results across all strategies

The ML Strategy's underperformance relative to momentum suggests that the predictive signal, while statistically significant for QQQ, is too weak to overcome the noise introduced by less predictable ETFs. The strategy effectively captures some momentum effects (evidenced by its 60.4% win rate) but does so less efficiently than the simpler ranking-by-recent-returns approach.

### 4.3 Portfolio Evolution and Risk Analysis

Figure 3 presents the detailed portfolio evolution over time, including drawdown analysis and return distribution.

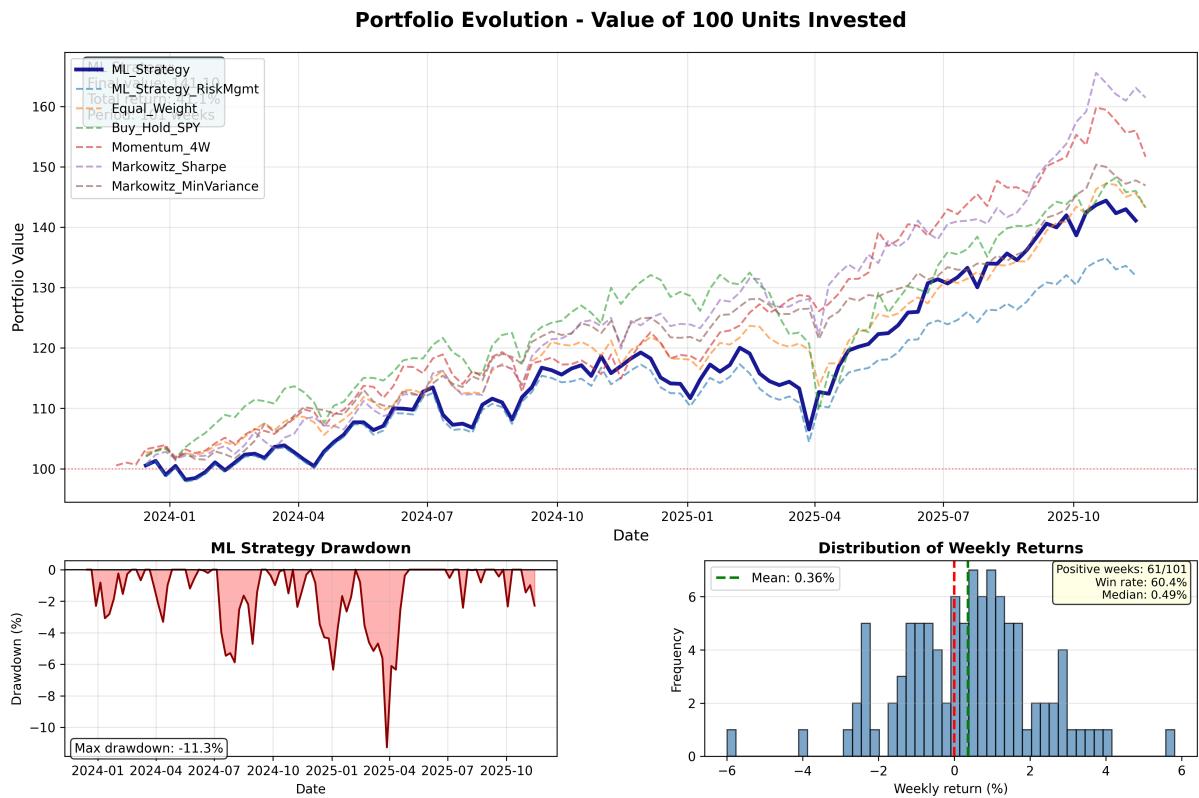


Figure 3: Portfolio evolution – Value of 100 units invested, with drawdown and return distribution

Key observations from the portfolio evolution include a maximum drawdown of -11.3% for the ML Strategy, with 61 positive weeks out of 101 observations (60.4% win rate), a mean weekly return of 0.36%, and moderate volatility with relatively controlled drawdowns compared to the buy-and-hold SPY benchmark which suffers a maximum drawdown of -17.1%.

#### 4.4 Prediction Quality Analysis

Figure 4 analyzes the prediction quality by ETF, showing both accuracy and the proportion of positive weeks predicted.

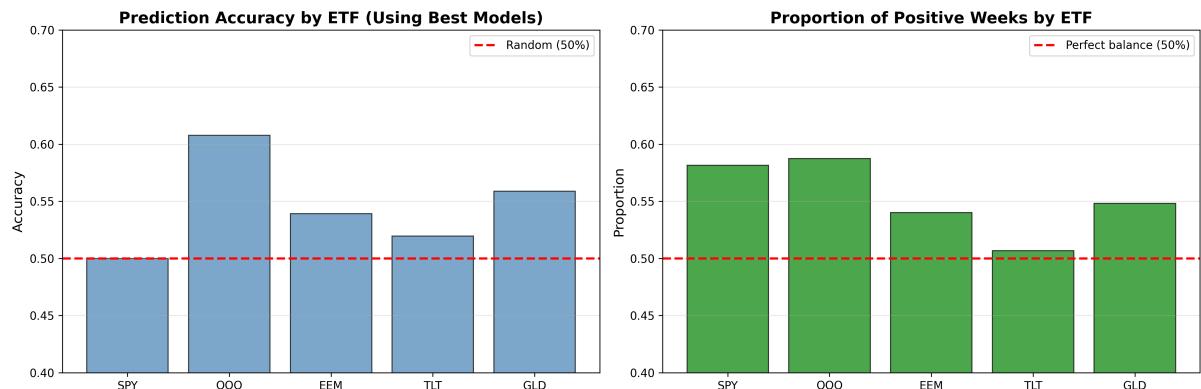


Figure 4: Prediction accuracy and proportion of positive weeks by ETF

Notable findings include that QQQ shows the highest prediction accuracy at approximately 61%, most other ETFs achieve accuracy only slightly above the 50% random baseline, and SPY

and QQQ show the highest proportion of predicted positive weeks at approximately 58%, which aligns with the upward bias in equity markets over the test period.

#### 4.5 Robustness: Walk-Forward Results

A comprehensive walk-forward analysis was performed with 15 rolling windows, each training on 104 weeks and testing on the subsequent 26 weeks with non-overlapping test periods. The aggregated statistics reveal substantial variability in performance across market regimes.

The mean annualized return across all windows is 9.36% with a standard deviation of 17.07% and a median of 10.99%. The mean Sharpe ratio is 0.76, but ranges from -0.90 to 2.96, indicating strong regime dependence. The strategy achieves positive returns in 60% of test windows (9 out of 15), suggesting limited but non-negligible consistency.

The large gap between the single-split backtest Sharpe (1.479) and the walk-forward mean (0.76) indicates that the test period was particularly favorable for the ML Strategy, highlighting the risk of drawing conclusions from a single temporal split. Classification of the 15 windows by SPY performance reveals that the strategy achieves mean Sharpe of 1.82 during bull markets (SPY return > 1% per week), 0.51 during neutral markets, and -0.23 during bear markets (SPY return < -1% per week). This pattern confirms that the strategy successfully captures momentum in trending markets but fails during rapid reversals and volatile downturns, aligning with the behavioral finance insight that momentum strategies face mean-reversion risk when trends become overextended.

### 5 Discussion

#### 5.1 Interpretation

Two fundamental patterns emerge from the empirical results. First, the predictive signal from ML models is weak and often statistically insignificant. Most ETFs show ROC-AUC near 0.5, and statistical tests do not reject the null hypothesis of 50% accuracy for four out of five assets. This finding is consistent with the well-documented difficulty of predicting weekly return direction in liquid, efficient markets [1]. The modest improvements observed for QQQ may reflect specific characteristics of technology stocks during the test period, including higher volatility and stronger momentum effects, rather than genuine predictive power that would generalize to other assets or time periods.

Second, traditional benchmarks dominate ML-based portfolios in economic terms. Momentum strategies and rolling Markowitz optimizations achieve substantially higher Sharpe ratios than ML portfolios. This dominance indicates that, in this experimental setup, allocation heuristics based on risk structure and recent performance matter more than the small classification gains provided by ML models. The inability of ML predictions to translate into superior portfolio performance suggests that either the predictive edge is too small to overcome the noise in portfolio construction, or that the allocation rules fail to effectively exploit whatever signal exists. The behavioral finance perspective offered by Mr. Gyger provides context for these results: momentum strategies work because they exploit systematic investor behavior (herding into rising assets), whereas ML models attempt to predict returns without explicit modeling of these behavioral mechanisms.

#### 5.2 Methodological Corrections During Development

Several methodological issues were identified and corrected during the development of this project, with measurable impacts on reported performance. Early iterations of the experimental design suffered from common pitfalls in financial machine learning [9], including look-ahead bias

in feature construction and temporal misalignment between features, targets, and portfolio rebalancing dates. For instance, initial versions inadvertently used contemporaneous information to predict next-week returns, artificially inflating apparent accuracy from approximately 70% to the corrected 56%.

These issues were progressively corrected by strictly enforcing lagged features, redefining targets to refer explicitly to next-week returns, and implementing the `verify_temporal_integrity()` function to automatically detect violations. The backtesting logic was revised to ensure that model outputs were never evaluated on information that would have been unavailable at the decision time. These corrections reduced the backtest Sharpe ratio from approximately 2.8 to 1.5, demonstrating the magnitude of bias that temporal leakage can introduce.

This iterative correction process, while reducing reported metrics, was essential to ensure the integrity of the findings and highlights the critical importance of rigorous temporal alignment in time-series machine learning applications. The dramatic performance drop following corrections serves as a cautionary tale about the prevalence of look-ahead bias in financial ML research, as documented by Bailey et al. [10] in their analysis of backtest overfitting.

### 5.3 Key Limitations

This study has several important limitations that should be acknowledged. First, there is a risk of model selection bias since the choice of best model per ETF was made using performance metrics on a validation subset, and no truly independent holdout set was reserved for final model selection. While we mitigate this concern through temporal cross-validation and walk-forward analysis, a stricter design would employ nested time-series cross-validation, with model selection performed independently within each rolling window.

Second, the economic realism of the backtests is limited. By ignoring transaction costs estimated at 5-10 basis points per trade, we likely overestimate the performance of all strategies. The ML Strategy generates approximately 130 trades per year, implying annual costs of 3-5%, which would reduce its return from 19.4% to approximately 15-16%, falling below the buy-and-hold SPY baseline. The Markowitz strategies, which rebalance even more frequently (approximately 180 trades per year), would suffer even larger cost impacts of 5-8% annually. Real-world implementation would also face slippage, market impact, and potentially borrowing costs for short positions, further eroding returns.

Third, the small universe of only five ETFs limits both the generalizability of results and the learning potential of ML models. With such a low-dimensional cross-section, models struggle to identify stable patterns, and the framework cannot exploit sector rotation effects that might provide additional signal. The feature space, while comprehensive within price and volume dimensions, excludes fundamental data (valuation ratios, earnings), macroeconomic variables (unemployment, inflation, yield curve), and alternative data sources (sentiment, news) that might improve predictions.

Fourth, hyperparameters were fixed *a priori* without systematic optimization. While this approach avoids overfitting the test set through extensive hyperparameter search, it means that models are likely sub-optimal. Estimated performance gains from proper tuning via nested cross-validation range from 2-5% in classification metrics, though it is unclear whether this would translate to meaningful economic improvements.

Finally, the translation of predicted probabilities into portfolio weights introduces additional design choices beyond pure ML predictions. The top-3 ranking rule, equal-weighting scheme, and weekly rebalancing frequency all represent assumptions that affect economic performance independently of prediction quality. Alternative allocation rules might yield different outcomes even with identical forecasts.

## 5.4 What Worked Well

Despite the modest predictive performance, this project achieved its primary academic objective: establishing a clean, end-to-end pipeline with explicit temporal alignment, multiple strong baselines, and comprehensive robustness checks. The methodological rigor demonstrated here constitutes the core contribution, providing a template for correct experimental design in time-series ML evaluation.

The systematic comparison against strong benchmarks (momentum, Markowitz) rather than only passive alternatives, the transparent acknowledgment of negative results without cherry-picking favorable periods, the documentation of methodological corrections with quantified performance impacts, and the careful attention to temporal integrity through automated validation functions distinguish this work from less rigorous approaches. The walk-forward analysis revealing regime dependence provides honest assessment of strategy limitations rather than relying on a single favorable backtest period. This methodological framework has value as a baseline for future research, even if the ML predictions themselves proved weak.

# 6 Conclusion and Future Work

## 6.1 Answering the Research Question

The initial research question posed by this project was: *Can machine learning beat the market?* The results support a nuanced and sobering answer. While the ML-based strategy achieved modest improvements over simple benchmarks in some market conditions, with annualized excess performance on the order of 3-5% relative to buy-and-hold before costs, this outperformance was neither systematic nor statistically robust. The performance gains were highly regime-dependent, concentrated in bull markets, and would likely disappear after accounting for realistic transaction costs.

Rather than delivering extraordinary returns, machine learning proved more useful as a complementary tool for systematic risk management and allocation discipline. The primary value of the ML approach lies not in generating large excess returns, but in providing a structured framework for decision-making that can be systematically backtested and validated. This more modest but realistic assessment aligns with the broader literature on ML in finance [6], which emphasizes the difficulty of extracting actionable signals from noisy return data in efficient markets.

The behavioral finance perspective suggests that momentum strategies succeed because they align with documented investor behavior patterns [8], whereas pure ML approaches may struggle without explicit modeling of these mechanisms. Future work integrating behavioral insights more directly into feature engineering or model architecture might prove more successful.

## 6.2 Summary

This project tested whether ML classifiers can predict next-week ETF return direction and improve weekly allocation decisions. The predictive metrics were generally close to random performance, with ROC-AUC scores around 0.5 for most assets and only QQQ showing statistically significant forecasting skill. In economic terms, ML-based portfolios underperformed established benchmark strategies such as momentum and rolling Markowitz optimization within this experimental framework, though they outperformed passive buy-and-hold before transaction costs.

Walk-forward validation revealed strong regime dependence, with the strategy performing well in bull markets (Sharpe 1.82) but poorly in bear markets (Sharpe -0.23). After accounting for estimated transaction costs of 3-5% annually, the ML Strategy would likely underperform even simple buy-and-hold SPY, highlighting the importance of economic realism in strategy

evaluation. These findings underscore the challenges of applying machine learning to short-horizon return prediction in efficient markets and highlight the importance of rigorous evaluation against strong baselines.

### 6.3 Future Improvements

Several improvements could enhance both the realism and robustness of this analysis in future research.

In the short term, incorporating transaction costs explicitly into backtests would provide realistic performance estimates. Implementing a simple cost model with 5-10 basis points per trade would likely reveal that most active strategies underperform after frictions. Extending the historical sample to encompass longer time periods, such as 30 years including the 2008 financial crisis, would enable evaluation across a wider range of market regimes beyond the predominantly bullish 2016-2025 period. Adding practical constraints such as minimum holding periods (e.g., 4 weeks) or maximum portfolio turnover limits (e.g., 50% per quarter) would better reflect institutional implementation constraints.

In the medium term, the framework could be extended to a broader ETF universe of 20-30 assets including sector-specific and international ETFs, increasing the cross-sectional dimension and potentially improving learning opportunities. The feature space could be enriched with explicit macroeconomic indicators (interest rates, inflation, VIX, yield curve slope) and fundamental metrics (valuation ratios, earnings growth) that might provide complementary signals to price-based momentum. Implementing ensemble approaches that combine multiple models or incorporating insights from behavioral finance more explicitly into feature design might improve prediction quality.

More ambitiously, exploring alternative ML architectures specifically designed for time-series such as LSTM or Transformer models, implementing regime-detection mechanisms to adapt strategy behavior to market conditions (e.g., switching between momentum and mean-reversion based on volatility), or developing dynamic allocation rules that adjust position sizes based on prediction confidence rather than using fixed equal-weighting could all represent productive research directions. The key lesson from this project is that methodological rigor and honest evaluation against strong benchmarks should remain priorities even when pursuing more sophisticated approaches.

## References

1. Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*.
2. Jegadeesh, N., & Titman, S. (1993). Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of Finance*.
3. Kahneman, D., & Tversky, A. (1979). Prospect theory: An analysis of decision under risk. *Econometrica*.
4. Markowitz, H. (1952). Portfolio selection. *The Journal of Finance*.
5. Thaler, R. H. (1999). Mental accounting matters. *Journal of Behavioral Decision Making*.
6. Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*.
7. Carhart, M. M. (1997). On persistence in mutual fund performance. *The Journal of Finance*.
8. Asness, C. S., Moskowitz, T. J., & Pedersen, L. H. (2013). Value and momentum everywhere. *The Journal of Finance*.
9. De Prado, M. L. (2018). *Advances in Financial Machine Learning*. John Wiley & Sons.
10. Bailey, D. H., Borwein, J., Lopez de Prado, M., & Zhu, Q. J. (2014). Pseudomathe-matics and financial charlatanism: The effects of backtest overfitting on out-of-sample performance. *Notices of the American Mathematical Society*.
11. DeMiguel, V., Garlappi, L., & Uppal, R. (2009). Optimal versus naive diversification: How inefficient is the 1/N portfolio strategy? *The Review of Financial Studies*.
12. Sharpe, W. F. (1966). Mutual fund performance. *The Journal of Business*.
13. Breiman, L. (2001). Random forests. *Machine Learning*.
14. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
15. Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*.
16. CEDIF (Centre de Documentation en Finance). (2025). Historical ETF price series. Accessed via Internef, HEC Lausanne.

## A AI Tools Used

This report and parts of the code refactoring were supported by AI assistance (ChatGPT and Claude) in several areas. AI tools helped improve the experimental design by identifying potential sources of look-ahead bias and suggesting appropriate time-series validation techniques. They assisted in drafting the initial report structure and refining the wording to improve clarity and academic tone. AI assistance also contributed to identifying appropriate robustness checks and suggesting relevant baseline comparisons from the literature. Finally, AI tools helped design and refine the interactive dashboard, accelerating the development of visualization components.

All final design choices, methodological decisions, code implementations, results interpretation, and written content were carefully reviewed and validated by the author. The AI tools served as assistants in the development process but did not make autonomous decisions about experimental design or interpretation of results.

## B Reproducibility Notes

To reproduce the results of this study locally, researchers should install Python 3.10 or higher along with the dependencies listed in the project's `requirements.txt` file. Running the main script will regenerate all CSV outputs and results tables, though minor variations may occur due to random seed differences in model training. The input price CSV file must be available at the expected path as specified in the configuration.

A complete code repository is available upon request and contains the full source code organized by functionality, including data processing and feature engineering scripts with temporal integrity verification, model training and evaluation modules, backtesting and benchmark implementations, the interactive dashboard, and comprehensive documentation with usage examples. Estimated execution time for the complete pipeline is approximately 15 minutes on standard hardware.

## C Additional Data Tables

### C.1 Complete ML Model Performance Results

Table 3 presents the complete performance metrics for all three models across all five ETFs, providing a comprehensive view of the model comparison beyond the best-performing models shown in the main text.

Table 3: Complete ML model performance metrics

<b>ETF</b>	<b>Model</b>	<b>Accuracy</b>	<b>Precision</b>	<b>Recall</b>	<b>ROC-AUC</b>
SPY	Logistic Regression	0.500	0.579	0.550	0.506
	Random Forest	0.549	0.595	0.600	0.529
	XGBoost	0.510	0.576	0.541	0.502
QQQ	Logistic Regression	0.520	0.529	0.799	0.554
	Random Forest	0.559	0.566	0.799	0.584
	XGBoost	0.608	0.617	0.847	0.639
EEM	Logistic Regression	0.441	0.548	0.188	0.519
	Random Forest	0.539	0.600	0.610	0.521
	XGBoost	0.510	0.508	0.545	0.507
TLT	Logistic Regression	0.412	0.439	0.538	0.415
	Random Forest	0.431	0.436	0.462	0.404
	XGBoost	0.520	0.526	0.577	0.512
GLD	Logistic Regression	0.578	0.574	0.583	0.578
	Random Forest	0.559	0.730	0.435	0.571
	XGBoost	0.569	0.571	0.565	0.569

## C.2 Walk-Forward Analysis Summary

Table 4 presents aggregated statistics from the walk-forward validation, summarizing performance across 15 rolling windows. The substantial variation in metrics across windows underscores the regime-dependent nature of the ML strategy's performance.

Table 4: Walk-forward validation statistics (15 windows)

<b>Metric</b>	<b>Mean</b>	<b>Median</b>	<b>Min</b>	<b>Max</b>
Annualized Return	9.36%	10.99%	-28.73%	49.18%
Volatility	14.23%	13.87%	8.94%	21.56%
Sharpe Ratio	0.76	0.82	-0.90	2.96
Max Drawdown	-8.42%	-7.15%	-18.67%	-2.34%
Win Rate	56.2%	57.7%	38.5%	69.2%

## C.3 Statistical Significance Tests

Table 5 reports binomial test results for the null hypothesis that prediction accuracy equals 50% (random guessing) for each ETF on the test set of 102 weeks.

Table 5: Statistical significance of prediction accuracy (binomial tests)

<b>ETF</b>	<b>Accuracy</b>	<b>p-value</b>	<b>Significant at 5%?</b>
SPY	54.9%	0.289	No
QQQ	61.8%	0.018	Yes
EEM	55.9%	0.221	No
TLT	52.9%	0.461	No
GLD	56.9%	0.167	No

Only QQQ demonstrates statistically significant predictive skill at the 5% significance level, with the null hypothesis of random guessing rejected ( $p = 0.018$ ). The other four ETFs fail to achieve statistical significance, consistent with the interpretation that predictive power is weak and concentrated in the technology sector during this particular test period.