Predicting Post-Earnings Excess Returns: A Machine Learning Investigation

Ricardo Contente Guerreiro
Student No. 20418497
MSc Finance
HEC Lausanne

December 2025

Abstract

This study investigates whether machine learning models can predict 30-day post-earnings excess returns for S&P 500 stocks using pre-announcement fundamental and market data. We test the null hypothesis that post-earnings returns are not predictable from publicly available information. Using a large sample of earnings announcements from 2010 to 2024, we construct a dataset of pre-announcement predictors that summarize firms' accounting fundamentals and recent market dynamics. We evaluate several standard machine learning approaches against simple benchmark strategies within a strictly time-ordered framework designed to prevent look-ahead bias and information leakage. Across models and robustness checks, we find no economically meaningful out-of-sample predictability, thereby confirming our null hypothesis in this setting. The evidence is consistent with market efficiency and rapid price adjustment around earnings announcements for liquid large-cap equities. Beyond the empirical result, the project provides a transparent, fully reproducible workflow and a leakage-aware methodology for reliable out-of-sample evaluation in financial machine learning.

Keywords: earnings announcements, machine learning, return prediction, market efficiency, post-earnings drift, reproducibility

Contents

Glossary

S&P 500	Large-cap U.S. equity universe used as the study sample.
Fundamentals	Accounting-based firm indicators (profitability, growth, efficiency, leverage) available before the announcement.
Earnings surprise	The gap between reported earnings and what the market expected, often used as a proxy for the "news" content of an earnings announcement.
Post-earnings excess return	The stock's post-announcement performance measured relative to the market benchmark over the same window.
EMH	Efficient Market Hypothesis: prices rapidly reflect public information, leaving little predictable "free" return.
PEAD	Post-Earnings Announcement Drift: tendency for prices to keep moving after earnings news instead of adjusting instantly.
SPY benchmark	SPDR S&P 500 ETF used as a practical proxy for the market return.
CAPM-style benchmark	Baseline that predicts a stock's post-earnings performance from its exposure (beta) to market moves.
Historical average	Baseline that always predicts the average outcome observed in the training data.
CRSP	Standard database of U.S. stock prices and returns used for market data construction.
Compustat	Standard database of firms' financial statements used for accounting variables.
I/B/E/S	Database of analyst forecasts and reported earnings, commonly used to compute earnings surprises.
In-sample	Performance measured on data the model has already seen during training.
Out-of-sample	Performance measured on held-out data not used for training or tuning (the realistic test).
Leakage / look-ahead bias	Any use of information that would not have been known at prediction time, which can inflate results artificially.
Volatility spikes	Sudden short-lived jumps in volatility (price variability), often around announcements or shocks, which can destabilize return patterns.
Policy shifts	Changes in monetary/fiscal/regulatory policy that can alter market "regimes" and the relationship between predictors and returns.
Ridge regression	Regularized linear regression that is stable when predictors are correlated and signal is weak.
Logistic regression	Linear classification model that outputs probabilities for a binary outcome.
Tree-based ensembles	Non-linear models built from many decision trees (e.g., Random Forest, Gradient Boosting, XGBoost).
Hyperparameters	Model settings chosen by the researcher (e.g., depth, learning rate) and tuned on validation data.
PCA	Method that compresses many variables into a few components; avoided here because it reduces interpretability.
R^2	Regression metric for explained variation; values near zero indicate little out-of-sample explanatory power.
MAE	Average absolute prediction error (easy to interpret in the target's units).
RMSE	Error metric that penalizes large mistakes more than MAE.
ROC-AUC	Classification metric measuring how well the model ranks positives above negatives, independent of a threshold.

1 Introduction

Earnings announcements are key information events in financial markets, where publicly traded firms disclose quarterly performance and update investors' expectations. A fundamental question in financial economics is whether such disclosures—and more generally the *publicly available information available before the announcement*—can predict subsequent stock returns. This question directly relates to the semi-strong form of the Efficient Market Hypothesis (EMH), which implies that public information should be rapidly incorporated into asset prices[4]. In this project, we test this implication in an event-driven setting by predicting 30-trading-day post-earnings *excess returns* for S&P 500 stocks relative to SPY using strictly pre-announcement fundamentals and market indicators.

1.1 Motivation

Historical research has documented post-earnings announcement drift (PEAD), where firms with positive earnings news tend to exhibit continued abnormal performance over subsequent weeks or months[1]. This pattern is often interpreted as delayed price adjustment, although alternative explanations (e.g., compensation for risk or limits to arbitrage) have been discussed in the literature. More recent evidence suggests that the magnitude and exploitability of earnings-related anomalies may have weakened in modern, liquid markets characterized by faster information diffusion, broad analyst coverage, and algorithmic trading. Understanding whether machine learning can extract stable predictive signals from pre-announcement data therefore remains an open empirical question. Economically meaningful predictability in a large-cap universe would challenge market efficiency; however, the absence of meaningful out-of-sample predictability under strict evaluation is consistent with rapid incorporation of public information. We focus on a 30-trading-day horizon as a pragmatic compromise: long enough to capture any delayed adjustment, while limiting longer-horizon confounding shocks unrelated to the earnings event.

1.2 Research Question and Objectives

This study investigates whether machine learning can predict 30-day post-earnings excess returns for S&P 500 stocks using strictly pre-announcement fundamental and market data. Specifically, we predict the excess return of a stock relative to the SPY benchmark over the 30 trading days following an earnings announcement.

We formulate and test the null hypothesis H_0: *out-of-sample predictive performance is not meaningfully better than naive baselines when forecasting 30-day post-earnings excess returns from pre-announcement information.* Our objectives are to:

1. Construct a reproducible, leakage-aware ML pipeline with rigorous data-quality controls and time-consistent train/validation/test splits
2. Engineer interpretable features from fundamentals, market indicators, and momentum signals computed strictly pre-announcement
3. Compare multiple model families against strong naive baselines under proper out-of-sample evaluation
4. Validate conclusions using statistical hypothesis testing (e.g., permutation tests, bootstrap intervals) and robustness experiments across alternative horizons and time periods
5. Document methodological choices, limitations, and reproducibility protocols to support credible interpretation of null results

1.3 Scope and Contributions

This project focuses on S&P 500 stocks over 2010–2024, covering a long sample that spans multiple market environments. We deliberately restrict the universe to large, liquid equities to keep the setting economically realistic and computationally feasible, and to reduce microstructure-related noise that is more severe for illiquid securities.

To keep the empirical question well-posed, we also limit the information set to variables that are available strictly *before* the earnings announcement (fundamentals and market-based indicators). We do not attempt to incorporate intraday announcement reactions, transaction costs, or alternative information sources (e.g., analyst forecasts or textual data), which would require additional data access and methodological choices beyond the scope of this course project.

Our main contributions are:

1. A leakage-aware, time-consistent evaluation design that emphasizes credible out-of-sample inference rather than in-sample fit
2. Statistical validation of null results using complementary checks (e.g., permutation tests and bootstrap uncertainty quantification)
3. A fully reproducible workflow with documented code, configuration management, and transparent reporting

1.4 Report Organization

Section 2 reviews the literature on earnings-related return predictability and key methodological pitfalls in financial machine learning. Section 3 describes the data sources, feature engineering, models, and evaluation methodology. Section 4 reports empirical results for regression, classification, and robustness experiments. Section 5 discusses interpretation, limitations, and implications of the findings. Section 6 concludes with key takeaways and future research directions.

2 Literature Review

This section briefly reviews evidence on earnings-related return predictability and positions our event-driven test in a modern large-cap setting. We summarize EMH and post-earnings announcement drift (PEAD), then connect these insights to our strict pre-announcement feature design.

2.1 Earnings announcements, market efficiency, and PEAD

Under the semi-strong form of the Efficient Market Hypothesis (EMH), publicly available information should be rapidly incorporated into prices, implying limited predictability of post-announcement excess returns from pre-announcement data[4]. Yet classic event studies document post-earnings announcement drift (PEAD)[8], where returns continue to drift in the direction of earnings news over subsequent weeks or months[1 2]. Importantly, the interpretation is not unique: abnormal returns depend on the benchmark and risk adjustment, and drift may reflect under-reaction, risk premia, or limits to arbitrage.

For large, liquid, and well-covered firms, recent evidence suggests earnings-related anomalies may be weaker and harder to exploit, making the existence of a stable predictive signal an open empirical question in our setting.

2.2 Signals and datasets used in related studies

Studies on earnings-related predictability typically draw on three broad groups of predictors: (i) fundamentals and earnings-related variables (e.g., profitability, valuation ratios, earnings surprise/expectations), (ii) market-based signals (momentum, volatility, liquidity/volume), and (iii) firm characteristics (size, value, quality). Momentum effects over multi-month horizons are well documented[7], and liquidity/volume variables have been linked to short-horizon return dynamics in several settings[3].

Regarding data, much of the academic literature relies on established databases such as CRSP (prices/returns), Compustat (fundamentals), and I/B/E/S (analyst forecasts). These sources provide strong coverage and consistent definitions but are costly and often inaccessible outside academia. In contrast, projects based on public or semi-public data sources improve reproducibility but may require additional care regarding missingness, definitions, and corporate actions.

In this project, we follow these common signal families but restrict the information set to variables available strictly *before* the earnings announcement; we do not incorporate analyst forecast datasets or textual alternatives, which would introduce additional data requirements beyond the scope of this course project.

2.3 Machine learning for return prediction and common pitfalls

Machine learning has been applied to return prediction with mixed evidence on out-of-sample performance. Large-scale studies report that flexible models can extract modest predictive structure when using many firm characteristics, but results depend strongly on the evaluation design and the stability of relationships over time[5]. In event-driven settings such as earnings announcements, the main challenge is that small leakages around the event date or biased validation schemes can create misleading performance.

Common pitfalls are well documented:

1. **Data leakage**, especially when features inadvertently incorporate post-announcement information (directly or via window alignment).

2. **Inappropriate validation**, such as random splits for time-ordered data, which overstates generalization performance.

3. **Non-stationarity and regime shifts**, where relationships learned in one period do not hold in another.

4. **Multiple testing / data snooping**, where repeated model selection, hyperparameter search, or many specifications inflate false discoveries.

5. **Selection biases**, including survivorship or index membership effects that limit external validity.

Methodological critiques emphasize that credible financial ML must rely on strict out-of-sample testing, transparent baselines, and conservative interpretation of results[6]. These concerns motivate the disciplined evaluation design adopted in this project.

2.4 Positioning and gap addressed

The literature motivates two competing views around earnings announcements. On the one hand, PEAD suggests that delayed price adjustment may create short-horizon predictability after earnings. On the other hand, modern large-cap markets are highly liquid and information-rich, and many reported anomalies may weaken once evaluated under strict out-of-sample designs and realistic information sets.

This project positions itself at the intersection of these perspectives by testing a narrow, event-driven question in a disciplined machine learning framework: can strictly pre-announcement fundamentals and market indicators predict 30-trading-day post-earnings *excess returns* for S&P 500 stocks relative to SPY?

Our contribution is not to propose a new trading rule, but to provide a credible empirical test in this setting by emphasizing:

1. **Leakage-aware design**: features are constructed strictly pre-announcement and evaluation relies on time-consistent train/validation/test splits.
2. **Strong baselines and uncertainty**: models are benchmarked against naive predictors and results are supported by statistical checks and robustness experiments.
3. **Reproducibility and transparency**: code, configuration, and methodological choices are documented to support interpretation—including when results are null.

3 Methodology

This section describes the data construction, feature design, and evaluation-focused implementation choices of our prediction pipeline. Because event studies are highly sensitive to timing, we emphasize reproducibility, strict leakage prevention around announcement dates, and a genuinely out-of-sample evaluation design.[1]

3.1 Data Description

Data sources. We rely on two primary sources: (1) Capital IQ-style fundamental data for S&P 500 firms (earnings announcement dates, EPS, revenue, and accounting ratios), accessed via the WRDS/CEDIF platform[10]; and (2) daily equity prices for individual stocks and the SPY benchmark retrieved from Yahoo Finance via the `yfinance` library[11]. The earnings dataset covers 2010 to Q4 2024 and contains 19,300 announcement events after filtering (95.2% retention from an initial 20,281 events).

Universe and sample construction. The universe consists of S&P 500 constituents. The test set contains 340 unique tickers. We acknowledge potential survivorship bias because index membership is based on current or recent constituents rather than historical membership. This reflects data availability constraints but limits generalizability to firms that survived the sample period.

Target variable. The primary regression target is the 30-day post-earnings excess return:

$$y_i = R_{i,[t,t+30]} - R^{\text{SPY}}_{[t,t+30]}, \tag{1}$$

where t is the earnings announcement date for event i, $R_{i,[t,t+30]}$ is the cumulative stock return over the subsequent 30 trading days, and $R^{\text{SPY}}_{[t,t+30]}$ is the corresponding SPY return. For classification experiments, we define a binary label:

$$y_i^{\text{binary}} = \mathbb{1}(y_i > 0),$$

indicating whether the stock outperformed the benchmark over the horizon.

[1]All features are computed using information available strictly before the earnings announcement.

Feature engineering. We engineer 21 interpretable features using strictly pre-announcement information, grouped as follows:

- **Fundamentals (15 features):** earnings surprise (percentage and value), profitability (ROE, ROA, net margin, operating margin), growth (revenue growth QoQ/YoY; net income change QoQ/YoY), efficiency (asset turnover; cash-to-assets), and capital structure (leverage; equity ratio; free cash flow proxy).
- **Market indicators (6 features):** pre-announcement volatility (30-day), trading volume (30-day average), SPY benchmark return (30-day), and stock momentum (1-month, 3-month, 6-month returns).

We prioritize interpretability and therefore avoid dimensionality reduction (e.g., PCA). While PCA can reduce noise and collinearity, its components are difficult to map to economic mechanisms and can obscure which pre-announcement signals drive predictions, which is central to our event-study setting.

Data quality and missingness. Missing values represent 23.2% of all feature cells, with higher rates for fundamental variables (e.g., EPS surprise: 42.9%, revenue growth YoY: 41.3%) than for market features ($< 2\%$). To avoid leakage, imputation parameters are fit on the training portion of each split (or fold) and applied unchanged to the corresponding validation/test data. We use median imputation as a simple and robust default, since it is less sensitive to outliers than mean imputation.

3.2 Approach

Baselines. We establish performance floors using two naive predictors: (i) a mean predictor that forecasts the historical average excess return, and (ii) a CAPM-style benchmark. These baselines ensure that machine learning models must demonstrate incremental out-of-sample value beyond simple heuristics.

Machine learning models. We use a small set of standard models for tabular data to balance interpretability and flexibility. For regression, we train Ridge regression and tree-based ensembles (Random Forest, XGBoost, and scikit-learn Gradient Boosting) to capture potential non-linearities and interactions. For classification, we train Logistic Regression, Random Forest Classifier, and Gradient Boosting Classifier. Hyperparameters are selected conservatively via grid search on the validation set to reduce overfitting risk.

Temporal train/validation/test splits. We use chronological splits to respect the time-ordered nature of earnings events and avoid look-ahead. Using a 2020-01-01 cutoff, we define a training set (9,808 events; 2010-02-03 to 2018-02-05), a validation set (2,452 events; 2018-02-05 to 2019-12-20), and a held-out test set (7,040 events; 2020-01-08 to 2025-03-13). The test set is never used during model fitting or hyperparameter selection.

Preprocessing. Features are standardized using `StandardScaler` fitted on the training set and applied unchanged to validation and test data. Missing values are imputed using the training-set median (applied unchanged out-of-sample) to prevent information leakage. We perform no feature selection in the main pipeline to preserve interpretability.

Evaluation metrics. For regression, we report R^2, MAE, and RMSE. For classification, we report ROC-AUC (primary) and accuracy (for completeness). All metrics are computed on the held-out test set.

Statistical validation and robustness. To test H_0 (no predictive power), we implement (i) bootstrap confidence intervals to quantify uncertainty in performance estimates and (ii) permutation tests to assess whether performance exceeds chance. We also conduct 5-fold *time-series* cross-validation (blocked / walk-forward) and robustness experiments across alternative horizons (0, 5, 10, 30 days) and evaluation windows.

3.3 Implementation

Software stack. The pipeline is implemented in Python 3.10 using standard scientific libraries. Key dependencies include `pandas`, `numpy`, `scikit-learn`, `yfinance`, `scipy`, `statsmodels`, and `matplotlib`. Dependencies are managed via Conda (`environment.yml`) and pip (`requirements.txt`) to support consistent environments across machines.

Repository structure. The project follows a modular 22-step architecture organized under `src/`, with each step implemented as a standalone Python module. The main entry point is `main.py`, which orchestrates the pipeline. Configuration (paths, random seeds, split dates) is centralized in `src/config.py`. Outputs are saved to `results/`, with subdirectories for each step.

Reproducibility and execution. All stochastic operations use `random_state=42` to support deterministic runs where possible. The complete pipeline can be executed from the project root via:

```
python3 main.py
```

Typical runtime is approximately 10 minutes on a standard laptop. Outputs include model metrics (JSON/CSV), publication-quality figures (PNG), and comprehensive logs.

Data snapshot note. Because Yahoo Finance data may be revised over time, downloaded price series are cached locally (with a documented download date). Reruns load the local snapshot by default unless a refresh is explicitly requested, reducing data-source drift and improving reproducibility.

Code example. Listing 1 illustrates the temporal split logic used to prevent look-ahead bias. from `src/preprocessing/dataset.py`:

```python
def train_val_test_split_time_based(
    df: pd.DataFrame,
    test_cutoff_date: str = "2020-01-01",
    val_fraction: float = 0.2
) -> Tuple[pd.DataFrame, pd.DataFrame, pd.DataFrame]:
    """
    Split data temporally to prevent look-ahead bias.
    """
    df_sorted = df.sort_values("earnings_date")

    test_mask = df_sorted["earnings_date"] >= test_cutoff_date
    df_test = df_sorted[test_mask].copy()
    df_pre_test = df_sorted[~test_mask].copy()

    n_val = int(len(df_pre_test) * val_fraction)
    df_val = df_pre_test.iloc[-n_val:].copy()
    df_train = df_pre_test.iloc[:-n_val].copy()

    return df_train, df_val, df_test
```

Listing 1: Temporal train/validation/test split to prevent look-ahead bias

4 Results

This section reports the empirical findings from our prediction pipeline. We focus on strict out-of-sample performance on the held-out test set and interpret results relative to naive baselines. For regression, the primary metric is test R^2 (with MAE/RMSE as scale-dependent complements); for classification, we report ROC-AUC (primary) and accuracy (for completeness). We then present visual diagnostics to assess whether any signal is visible beyond noise.

4.1 Experimental Setup

Evaluation design. All results are computed on a held-out test set (7,040 events; 2020-01-08 to 2025-03-13) that is never used during model training or hyperparameter selection. The validation set (2,452 events; 2018-02-05 to 2019-12-20) is used exclusively for grid search, ensuring a strict out-of-sample evaluation protocol.

Training protocol and preprocessing. For reproducibility, all models use `random_state=42`. Features are standardized using `StandardScaler` fitted only on the training set and applied unchanged to validation and test. Missing values (23.2% of feature cells) are imputed with the training-set median to prevent information leakage.

Software environment. Experiments are executed in a fixed Python 3.10 environment (see Section 3.3 for dependency management and versioning). All results are reproducible via `python3 main.py`.

4.2 Performance Evaluation

Table 1 reports regression performance on the held-out test set. All models yield test R^2 close to zero (and sometimes negative), meaning they explain essentially none of the variation in 30-day excess returns and do not improve materially over the naive baselines.

Model	**Test MAE**	**Test RMSE**	**Test R^2**
Baseline (Mean)	0.0554	0.0788	-0.0003
Baseline (CAPM)	0.0576	0.0818	-0.0786
Ridge	0.0557	0.0792	-0.0105
Random Forest	**0.0553**	**0.0786**	**0.0036**
XGBoost	0.0557	0.0793	-0.0149

Table 1: Regression performance on the out-of-sample test set (7,040 events). Random Forest attains the highest test R^2 (= 0.0036), explaining less than 1% of variance. Differences in MAE/RMSE relative to the mean baseline are negligible, and negative R^2 values indicate performance worse than predicting the mean. Overall, results do not reject H_0.

CAPM baseline. We construct a CAPM-style benchmark that predicts excess returns as $\hat{y}_{i,t} = \hat{\beta}_{i,t} R^{\text{SPY}}_{[t,t+30]}$, where $\hat{\beta}_{i,t}$ is estimated from a pre-announcement rolling window of daily returns (training-period data only).

Interpretation. The best-performing model (Random Forest) achieves test $R^2 = 0.0036$, i.e., less than 1% of variance explained. Ridge and XGBoost exhibit negative R^2, meaning they perform worse than the mean predictor out-of-sample. Overall, we find no evidence of economically meaningful predictability of 30-day post-earnings excess returns from pre-announcement information in our setting.

Table 2 reports classification performance. All models achieve ROC-AUC close to 0.50, indicating near-random discrimination between stocks that outperform vs. underperform the benchmark.

Model	**Test AUC**	**Test Accuracy**
Dummy (Most Frequent)	0.5000	0.4979
Logistic Regression	0.4991	0.4925
Random Forest Classifier	0.5134	0.5121
Gradient Boosting Classifier	**0.5138**	0.5091

Table 2: Classification performance on the test set. Best AUC is 0.514 (Gradient Boosting), only marginally above random guessing (0.50), suggesting no reliable directional predictability.

Statistical validation. For the best regression model (Random Forest), the permutation test rejects the strict null of zero predictive performance with $p < 0.001$ (1,000 permutations). However, the bootstrap 95% confidence interval for test R^2 is $[-0.0017, 0.0087]$, which includes zero, and the point estimate remains economically negligible ($R^2 = 0.0036$). We therefore interpret the effect as statistically detectable yet not robust enough to imply a reliable predictive edge.

Across robustness experiments over alternative horizons (0, 5, 10, and 30 trading days), the overall conclusion remains unchanged: models do not deliver economically meaningful out-of-sample predictability in our setting.

4.3 Visualizations

Figure 1 plots realized versus predicted 30-day excess returns for the Random Forest model on the held-out test set. The cloud of points forms a narrow horizontal band around zero on the y-axis: predictions exhibit very low dispersion and remain close to the unconditional mean even when realized outcomes vary widely. This shrinkage toward zero helps explain the near-zero out-of-sample R^2 and indicates that the model does not learn a stable mapping from pre-announcement features to post-earnings performance at the 30-day horizon in our setting.

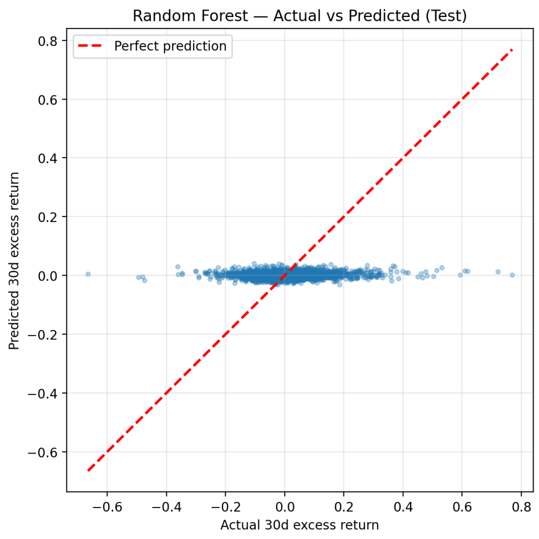


Figure 1: Random Forest (test set): actual vs. predicted 30-day excess returns. Predicted values are tightly clustered around zero across the full range of realized returns, indicating a near-mean predictor and consistent with negligible out-of-sample fit.

Figure 2 provides a compact overview of out-of-sample performance across all models. Across both regression and classification, results are tightly concentrated around the naive baselines: test R^2 values remain near zero and MAE/RMSE differences are negligible in magnitude. Even the best model (Random Forest) achieves only a marginal improvement ($R^2 = 0.0036$), which is economically tiny. Overall, the consistency of these outcomes across model families supports the conclusion that any exploitable signal from strictly pre-announcement features is, at best, extremely weak in our setting.

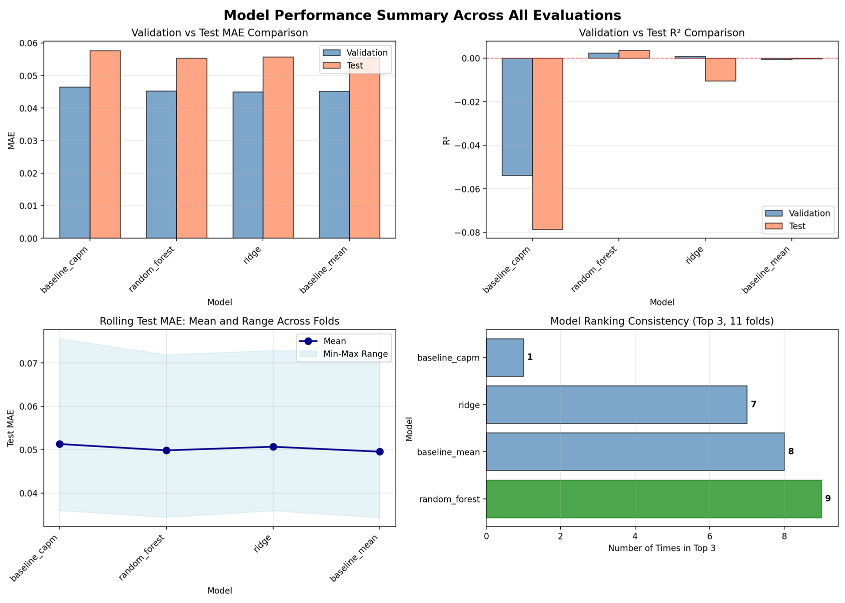


Figure 2: Test-set performance summary across models. All approaches cluster near $R^2 \approx 0$ with minimal variation in MAE/RMSE, indicating no economically meaningful improvement over naive baselines. Random Forest attains the highest test R^2 (0.0036), but the effect remains negligible.

5 Discussion

This section interprets the empirical findings, clarifies what the results do (and do not) imply about market efficiency, and discusses why predictability may be weak in this setting. We then outline limitations and feasible extensions that are most likely to change the signal-to-noise ratio rather than simply increasing model complexity.

5.1 Interpretation and link to the null hypothesis

Overall, the results support the null hypothesis H_0 from Section 1.2: models do not deliver *economically meaningful* out-of-sample improvements over naive baselines when predicting 30-day post-earnings excess returns from strictly pre-announcement information.

The best regression model (Random Forest) achieves test $R^2 = 0.0036$ (i.e., less than 1% variance explained). A permutation test rejects the strict null of zero performance ($p < 0.001$, 1,000 permutations), but the bootstrap 95% confidence interval includes zero ($[-0.0017,\ 0.0087]$). We therefore interpret the effect as statistically detectable yet economically negligible and not robust enough to imply a reliable predictive edge. Classification results are consistent with this view: the best ROC-AUC is approximately 0.51, only marginally above random guessing.

More broadly, these findings illustrate that greater model complexity does not guarantee better generalization in finance. In a liquid, well-covered large-cap universe, the lack of meaningful predictability is consistent with rapid incorporation of public information, in line with the market-efficiency perspective emphasized in our Investment course material.

5.2 Why might predictability be weak here?

Several complementary mechanisms can explain the near-zero out-of-sample performance in our setting:

1. **Low signal-to-noise at the 30-day horizon.** Over 30 trading days, excess returns are heavily affected by idiosyncratic shocks and macro news unrelated to the earnings event, which can dominate any incremental earnings-related signal.
2. **Non-stationarity and regime dependence.** Relationships between pre-announcement indicators and post-event returns may vary across market regimes (e.g., volatility spikes, policy shifts), limiting the transferability of patterns learned on earlier periods to later ones.
3. **Event-time alignment noise.** Earnings announcements may occur after-hours or before the open. If announcement timestamps are imperfectly aligned with the return measurement window, the target can embed additional noise that reduces learnability.
4. **Feature timing and measurement error.** Fundamental variables may be missing, updated with lags, or inconsistently defined across sources; substantial imputation can further dilute weak signals when missingness is non-random.
5. **Selection effects from a large-cap universe.** Restricting to S&P 500 constituents focuses on liquid, well-covered firms where information frictions may be smaller, leaving less room for systematic mispricing.
6. **Benchmark choice.** Using excess returns relative to SPY removes broad market movements. While appropriate for isolating relative performance, this choice can reduce apparent predictability compared with alternative benchmarks (e.g., sector- or factor-based).

5.3 Robustness, limitations, and implications

Robustness. Overall performance remains extremely small out-of-sample. The best regression model (Random Forest) achieves test $R^2 = 0.0036$ with a bootstrap 95% CI of $[-0.0017,\ 0.0087]$ (includes zero). A permutation test rejects the strict null of zero performance ($p < 0.001$, 1,000 permutations), but the effect size is economically negligible and does not translate into a reliable improvement over simple baselines.

Limitations and implications. Results should be interpreted in light of three constraints: (i) a large-cap, well-covered S&P 500 universe where mispricing opportunities may be limited; (ii) a 30-trading-day horizon that amplifies noise unrelated to the earnings event; and (iii) measurement issues (missing fundamentals, daily data granularity, and event-time alignment). In practical terms, these findings caution against expecting consistent post-earnings signals from standard pre-announcement features in this setting, and they highlight the importance of strict temporal validation and transparent baselines in financial ML.

5.4 Future directions

Future work should prioritize extensions that plausibly improve the signal-to-noise ratio while preserving strict temporal evaluation. Shorter event windows (1–10 trading days) may better capture any localized post-earnings drift and reduce contamination from unrelated news. Incorporating announcement timestamps (after-hours vs. before-open) could improve event-time alignment and reduce label noise. Richer earnings-specific information (e.g., textual features from earnings calls) may add incremental predictive content beyond standard fundamentals. Finally, sector- or factor-adjusted benchmarks and broader universes (mid/small caps) would test whether predictability exists outside large, highly efficient markets.

6 Conclusion and Future Work

This project asked a focused question: can machine learning predict 30-day post-earnings *excess returns* for S&P 500 stocks using only information available *before* the earnings announcement? To answer it, we built a reproducible and leakage-aware pipeline that respects event timing, relies on interpretable features grounded in fundamentals and recent market dynamics, and evaluates models under strict time-ordered splits against strong naive baselines.

The main empirical takeaway is simple: in this setting, we do not find *economically meaningful* out-of-sample predictability. While small statistical differences can occasionally appear depending on the specification, they remain too weak and unstable to constitute a reliable signal at the 30-day horizon. In practical terms, the information contained in standard public fundamentals and pre-announcement price/volume patterns does not translate into consistent, tradable forecasting power for large-cap, liquid equities.

More importantly, the project was a learning exercise in what makes financial prediction difficult. The work highlights how easily apparent predictability can be created by subtle design mistakes (timing mismatches, leakage around event dates, overly flexible tuning), and how disciplined evaluation can turn an attractive hypothesis into a more realistic conclusion. It also illustrates a broader point: in high-noise environments such as stock returns, a well-executed "negative" result is informative because it clarifies what does *not* work under a clearly defined information set and protocol.

Several limitations frame the scope of this conclusion. First, the universe (S&P 500) is intentionally efficient and highly liquid, which may mechanically reduce predictable patterns. Second, our features are deliberately conservative and mostly drawn from standard public data; richer information channels (text, options, order flow, investor attention) are outside the current scope. Third, the 30-day horizon is a demanding target: any earnings-related signal may be strongest immediately after the event and fade quickly. Finally, our design choices prioritize transparency and robustness over maximum complexity; alternative modeling choices could be explored, but would need to preserve the same leakage-aware discipline.

Future work can therefore extend the project along several natural directions. A first step is to test shorter and more event-focused horizons (e.g., 1–10 trading days) and refine event-time alignment (after-hours vs. before-open) to better match how information reaches markets. A second direction is *regime analysis*: the 2020 COVID period is a compelling case study to examine whether predictability changes during extreme shocks, and whether market incorporation differs during crash and recovery phases. A third direction is to expand the information set toward what real-world practitioners monitor: analyst expectations and revisions, earnings-call language, news and social-media signals, and potentially options-implied measures of uncertainty. Finally, broadening or narrowing the sample (beyond the S&P 500, different liquidity tiers, sector subsamples) could help identify where predictability is more plausible and where market efficiency is most binding.

Overall, this project does not "reinvent the wheel"—and that is precisely the point. It provides a careful, transparent, and reproducible test of a realistic hypothesis under strict evaluation. The conclusion is not that prediction is impossible in general, but that for liquid large-cap equities and standard public pre-announcement information, sophisticated models may still fail to extract economically meaningful signals unless the horizon, information set, or market regime fundamentally changes.

References

Academic Literature

[1] Ball, R., & Brown, P. (1968). An empirical evaluation of accounting income numbers. *Journal of Accounting Research, 6*(2), 159–178.

[2] Bernard, V. L., & Thomas, J. K. (1989). Post-earnings-announcement drift: Delayed price response or risk premium? *Journal of Accounting Research, 27*, 1–36.

[3] Chordia, T., Roll, R., & Subrahmanyam, A. (2001). Market liquidity and trading activity. *The Journal of Finance, 56*(2), 501–530.

[4] Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance, 25*(2), 383–417.

[5] Gu, S., Kelly, B., & Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies, 33*(5), 2223–2273.

[6] Harvey, C. R., Liu, Y., & Zhu, H. (2016). *... and the cross-section of expected returns. The Review of Financial Studies, 29*(1), 5–68.

[7] Jegadeesh, N., & Titman, S. (1993). Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of Finance, 48*(1), 65–91.

[8] Ke, B., & Ramalingegowda, S. (2005). Do institutional investors exploit the post-earnings announcement drift? *Journal of Accounting and Economics, 39*(1), 25–53.

Course Materials

[9] Goyal, A. (2024). *Portfolio strategies and market efficiency.* Investment Management course materials, HEC Lausanne (MSc Finance).

Data Sources

[10] Wharton Research Data Services (WRDS). (2024). *Compustat North America – Capital IQ fundamentals database.* Accessed via CEDIF platform (HEC Lausanne / University of Lausanne institutional access).

[11] Yahoo Finance. (2024). Historical market data for S&P 500 stocks and SPY ETF. Accessed via the `yfinance` Python library.

A Additional Results

This appendix provides supplementary diagnostics that support the main conclusions but are not essential to the primary narrative.

A.1 Cross-Validation Results

What this table shows. Table 3 reports 5-fold *time-series* cross-validation performance for Ridge and Random Forest (on the pre-test period). We summarize fold-level dispersion to assess stability across time blocks.

Key takeaway. Across folds, R^2 is unstable and confidence intervals include zero, consistent with weak or non-robust predictability. Absolute error metrics can differ from the held-out test set because folds cover earlier periods with different return dispersion.

Model	Metric	Mean	Std	Min	Max
	R^2	-0.117	0.176	-0.461	-0.003
Ridge	MAE	0.040	0.004	0.033	0.046
	RMSE	0.057	0.006	0.047	0.062
	R^2	-0.038	0.045	-0.123	0.011
Random Forest	MAE	0.041	0.005	0.034	0.046
	RMSE	0.055	0.006	0.048	0.063

Table 3: 5-fold time-series cross-validation summary (Ridge and Random Forest). Metrics are aggregated across folds. Negative R^2 indicates worse performance than predicting the mean within the corresponding fold.

A.2 Market Regime Analysis

What this table shows. Table 4 compares model performance across market regimes to test whether predictability varies with market conditions. We define *bull* vs. *bear* regimes using the sign of the SPY return over the same 30-trading-day horizon (ex post classification).

Key takeaway. Performance remains near-baseline in both regimes; differences in R^2 are economically negligible (on the order of 10^{-3}), suggesting that the null result is not driven by a single market environment.

Regime	Model	N (test)	R^2	MAE	RMSE
	Baseline	3,426	-0.001	0.056	0.080
Bull	Ridge	3,426	-0.029	0.056	0.081
	Random Forest	3,426	-0.022	0.056	0.080
	Baseline	3,614	0.000	0.055	0.078
Bear	Ridge	3,614	-0.009	0.055	0.078
	Random Forest	3,614	-0.002	0.055	0.078

Table 4: Model performance by market regime (test set). Regimes are defined by the sign of the realized SPY return over the same 30-day horizon.

A.3 Classification Model Performance

What this table shows. Table 5 reports classification metrics across train/validation/test splits for all classifiers.

Key takeaway. Test ROC-AUC values remain close to 0.50, indicating no reliable directional predictability. The Random Forest classifier exhibits clear overfitting (train AUC = 1.0), which further supports the conservative interpretation of near-random test performance.

Model	**Split**	**Accuracy**	**Precision**	**Recall**	**ROC-AUC**
Dummy (Most Frequent)	Train	0.522	0.522	1.000	0.500
	Val	0.535	0.535	1.000	0.500
	Test	0.498	0.498	1.000	0.500
Logistic Regression	Train	0.535	0.535	0.845	0.541
	Val	0.545	0.549	0.842	0.549
	Test	0.492	0.494	0.768	0.499
Random Forest	Train	1.000	1.000	1.000	1.000
	Val	0.524	0.550	0.612	0.521
	Test	0.512	0.509	0.571	0.513
Gradient Boosting	Train	0.642	0.624	0.794	0.714
	Val	0.540	0.557	0.697	0.543
	Test	0.509	0.505	0.649	0.514

Table 5: Classification metrics across all models and splits. Test ROC-AUC values close to 0.50 indicate near-random discrimination.

A.4 Supplementary Figures

This section presents additional diagnostic visualizations that support the robustness of our near-null findings.

Cross-validation stability. Figure 3 shows the distribution of R^2 scores across 5 time-series cross-validation folds for Ridge and Random Forest. Scores remain near zero (and often negative) across folds, indicating that the weak performance is not driven by a single split but is persistent across time blocks.

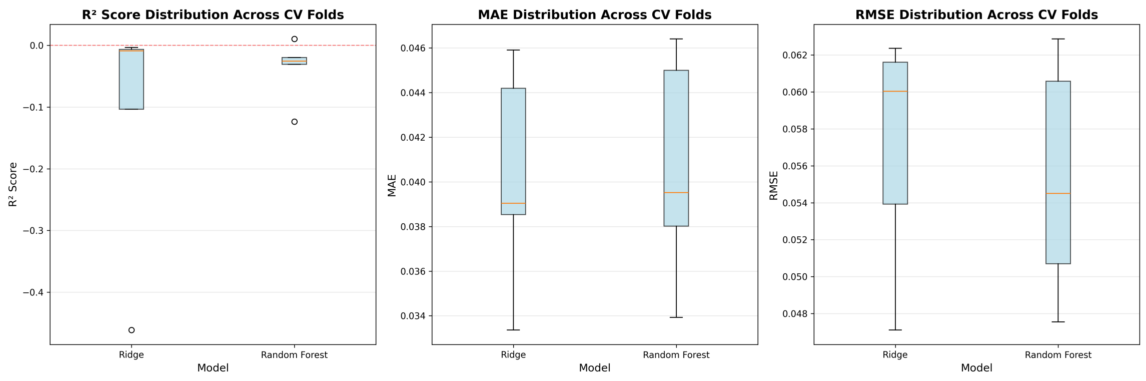


Figure 3: Cross-validation R^2 score distributions (5 folds) for Ridge and Random Forest.

Residual diagnostics (Ridge). Figure 4 plots residuals against predicted values for Ridge on the test set. The absence of visible structure suggests no obvious systematic bias, but the narrow range of predictions relative to the wide residual dispersion highlights the model's limited explanatory power.

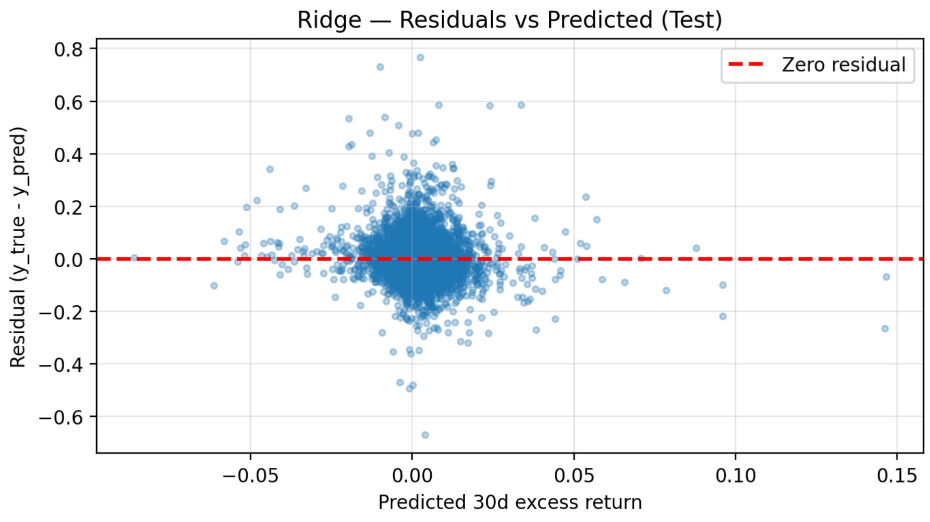


Figure 4: Ridge regression residuals vs. predicted values on the test set.

Feature importance (Random Forest). Figure 5 reports Random Forest feature importance rankings. Some market-related features (e.g., pre-announcement returns and volatility) receive higher importance scores; however, overall test performance remains near-null, suggesting that these features do not contain sufficiently stable signal to yield meaningful out-of-sample predictability.

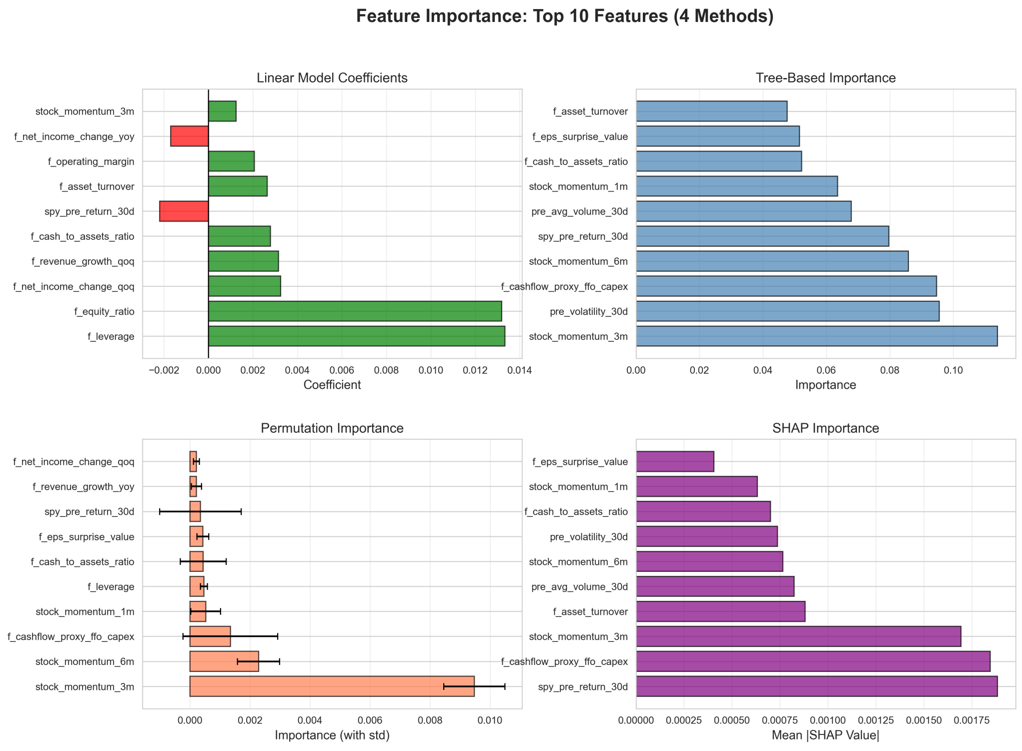
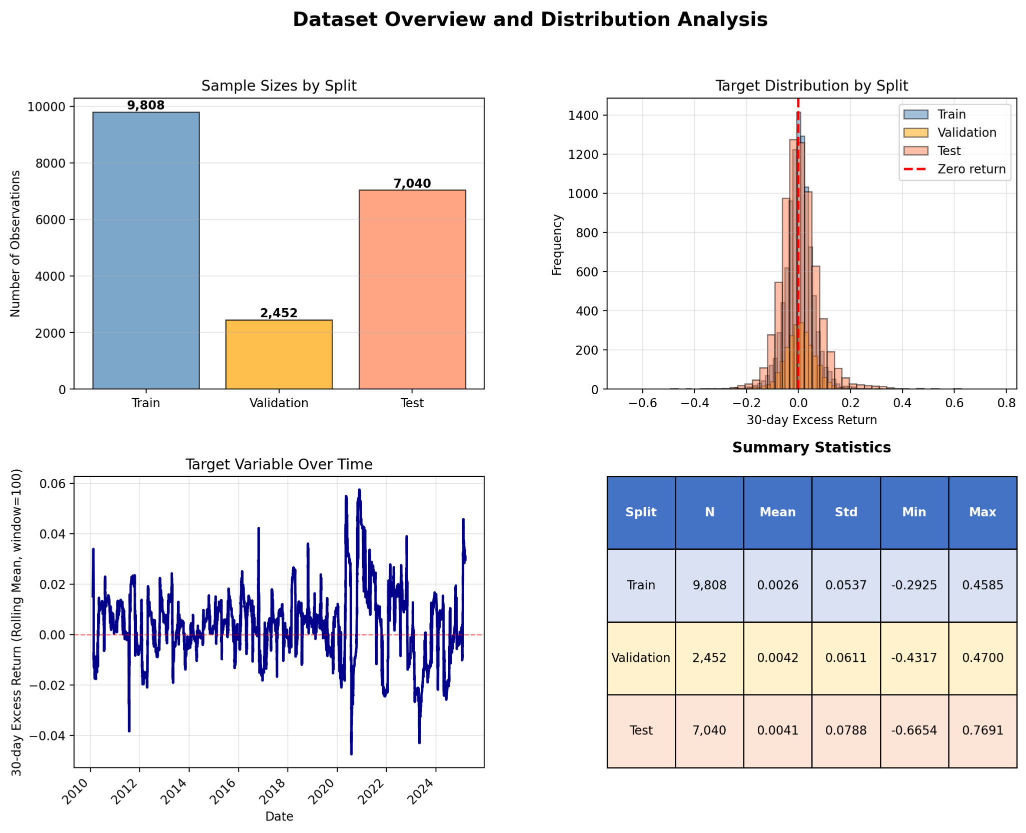


Figure 5: Random Forest feature importance rankings (test-set model).

Dataset overview and target distribution. Figure 6 summarizes the main descriptive properties of the dataset used in our experiments: sample sizes by split, the distribution of 30-day excess returns, and basic summary statistics. The target is centered close to zero with heavy tails, consistent with noisy excess-return prediction settings.



Split	N	Mean	Std	Min	Max
Train	9,808	0.0026	0.0537	-0.2925	0.4585
Validation	2,452	0.0042	0.0611	-0.4317	0.4700
Test	7,040	0.0041	0.0788	-0.6654	0.7691

Figure 6: Dataset overview and distribution analysis (train/validation/test split sizes, target distribution, and summary statistics).

ROC curves (test set). Figure 7 reports ROC curves for all classification models on the test set. Curves remain close to the 45-degree line and AUC values cluster around 0.50–0.514, indicating near-random discrimination between outperform vs. underperform events.

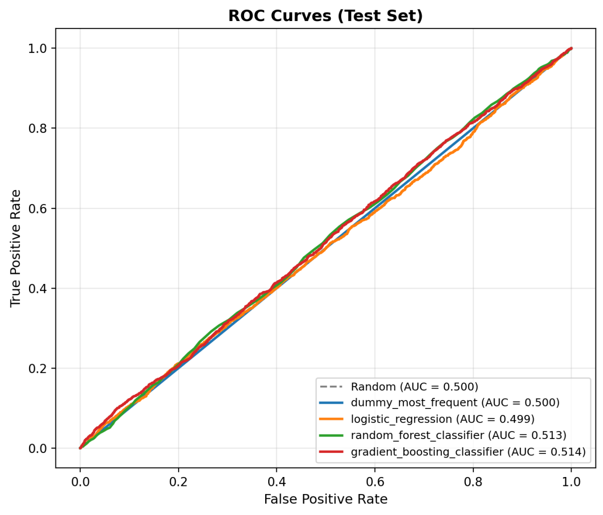


Figure 7: ROC curves on the test set for all classifiers. Performance is close to random (AUC ≈ 0.50).

B Code Repository

B.1 Repository Information

GitHub Repository: github.com/RicUnil/Linking-Pre-Fundamentals-to-Market-Reaction-using-ML

Project Structure:

```
Pre-Fundamentals-Project/
 README.md                    # Project documentation
 environment.yml              # Conda dependencies
 requirements.txt             # pip dependencies
 main.py                      # Single entry point (22-step pipeline)
 run_all_experiments.py       # Optional: runs all 5 experiments
 data/                        # Raw data files
    RAW_DATA.csv
    BENCHMARK.csv
    Quarter_*.csv
 src/                         # 22-step modular pipeline
    step_01_project_setup.py
    step_02_environment_setup.py
    ...
    step_22_data_quality_analysis.py
    config.py                # Configuration settings
    models/                  # Model implementations
    preprocessing/           # Feature engineering
    evaluation/              # Metrics and validation
    analysis/                # Statistical tests
    visualization/           # Plotting functions
 experiments/                 # 5 robustness experiments
    experiments_01/          # 10-day returns
    experiments_02/          # 5-day returns
    experiments_03/          # Day-0 reaction
    experiments_04/          # Window robustness
    experiments_05/          # Economic significance
 results/                     # Generated outputs
    figures/                 # All visualizations
    step_*/                  # Per-step results (JSON/CSV)
    experiments_*/           # Experiment outputs
 tests/                       # Unit tests
 notebooks/                   # Jupyter analysis notebooks
 processing/                  # Step-by-step summaries
 PROPOSAL.md                  # Project proposal
 AI_USAGE.md                  # AI tool usage documentation
```

B.2 Installation Instructions

Prerequisites. The project requires Python $\geq$ 3.10, a working `pip` installation or Conda/Miniconda, and an internet connection (to download market data). A standard laptop setup (e.g., 8 GB RAM) is sufficient.

```
# Clone repository
git clone https://github.com/RicUnil/Linking-Pre-Fundamentals-to-Market-
    Reaction-using-ML
cd Linking-Pre-Fundamentals-to-Market-Reaction-using-ML

# Create and activate environment
conda env create -f environment.yml
conda activate earnings-env
```

Listing 2: Conda setup (recommended)

```
# Clone repository
git clone https://github.com/RicUnil/Linking-Pre-Fundamentals-to-Market-
    Reaction-using-ML
cd Linking-Pre-Fundamentals-to-Market-Reaction-using-ML

# Create virtual environment
python3 -m venv .venv
source .venv/bin/activate  # Mac/Linux
# .venv\Scripts\activate   # Windows

# Install dependencies
pip install -r requirements.txt
```

Listing 3: venv + pip setup (local development)

B.3 How to Run

Main pipeline (required). Run the full 22-step pipeline from the repository root:

```
python main.py
```

Listing 4: Run full pipeline

What it does.

- Loads and validates inputs (Steps 1–7).
- Engineers 21 pre-announcement features (Steps 8–10).
- Trains models (Ridge, Random Forest, XGBoost, Gradient Boosting) (Steps 11–13).
- Evaluates models, runs statistical tests, and generates figures (Steps 14–22).

Outputs. Metrics are saved as `.json`/`.csv` under `results/`, and figures are saved under `results/`

Robustness experiments (optional). Run all robustness experiments:

```
python run_all_experiments.py
```

Listing 5: Run all robustness experiments

Or run them individually:

```
python3 -m experiments.experiments_01.src.experiment_01_returns_10d
python3 -m experiments.experiments_02.src.experiment_02_returns_5d
python3 -m experiments.experiments_03.src.experiment_03_day0_reaction
python3 -m experiments.experiments_04.src.experiment_04_window_robustness
python3 -m experiments.experiments_05.src.experiment_05_economic_significance
```

Listing 6: Run experiments individually

Run individual steps (advanced). For debugging or partial execution, run pipeline steps directly:

```
python3 -m src.step_01_project_setup
python3 -m src.step_10_create_final_dataset
python3 -m src.step_16_advanced_analysis
# ...
```

Listing 7: Run selected pipeline steps

```
pytest tests/ -v
```

Listing 8: Run unit tests