Winner Stock Prediction as Decision-Aligned Multiclass Classification

Blanca Elvira Fernandez Mendez
Department of Mathematics and Statistics
York University
Toronto, Ontario M3J 1P3

Pavlos Gkasis
International Business University
Toronto, Ontario M5G 2K4

Jairo Diaz-Rodriguez
Department of Mathematics and Statistics
York University
Toronto, Ontario M3J 1P3
jdiazrod@yorku.ca

Abstract

We study short-horizon equity selection through a decision-aligned winner prediction task: given recent market history for a fixed universe, predict which asset will achieve the highest next-day return. This reframes return forecasting as a multiclass classification problem and provides a natural analogue to next-token prediction in sequence modeling. Using a decade of daily data for seven large-cap NASDAQ equities, we compare forecast-then-argmax pipelines with direct discriminative models, including classical classifiers and Transformer encoders. We find that models achieving the highest top-1 accuracy often do so by collapsing predictions onto a small subset of dominant assets, while sequence models trade accuracy for substantially broader class coverage. These results expose an accuracy–concentration tradeoff in relative-return forecasting and highlight the need for class-sensitive and diversity-aware evaluation beyond headline accuracy in Financial AI benchmarks.

1 Introduction

Forecasting financial markets remains a central challenge for both quantitative finance and machine learning, not only because of the economic value of accurate predictions, but also because market dynamics are noisy, non-stationary, and heavily influenced by latent factors. Classic arguments from the Efficient Market Hypothesis suggest that publicly available information should be rapidly incorporated into prices, limiting the predictability of excess returns [5, 15]. At the same time, practitioners routinely deploy statistical and machine learning models to extract weak but actionable structure from historical data. This tension continues to motivate careful empirical studies that clarify which modeling choices help in realistic settings, and which apparent gains arise from problem formulation or evaluation artifacts.

Most work in stock prediction focuses on forecasting prices or returns for a single asset, commonly as a regression task. In practice, however, investment decisions are often relative: capital is allocated across a set of candidates, and the key question becomes which asset is likely to outperform its peers over a given horizon. Motivated by this observation, we study a simple but decision-aligned formulation: given a fixed universe of stocks and a window of past observations, predict which stock will achieve the highest next-day return. Concretely, for each trading day we label the *winner* as the asset with the maximum realized daily return within the universe, and we train models to predict the next winner. This recasting transforms next-day forecasting into a multiclass classification

Preprint. Under review.

problem with a discrete outcome space, and it creates a natural connection to "next-token" prediction objectives used in modern sequence modeling [21, 4, 27].

This formulation has two practical advantages. First, it avoids sensitivity to the scale of returns and emphasizes relative ordering, which is often what matters for allocation decisions. Second, it enables direct comparison between two families of approaches: (i) *continuous forecasting pipelines* that predict per-asset returns and then select the argmax, and (ii) *direct classifiers* that map historical features to a winner label. The setting is also challenging in ways that mirror real financial prediction: winner labels are typically imbalanced (some assets win more frequently), market regimes shift over time, and short-horizon returns exhibit heavy tails. As a result, model performance cannot be fully understood from accuracy alone; it is also important to characterize whether models collapse to a small subset of classes, a behavior that may be undesirable in downstream decision systems.

We evaluate representative methods from classical time series modeling, standard machine learning, and deep sequence modeling. For statistical baselines we consider autoregressive approaches, including ARIMA-style models and multivariate extensions [16, 26]. For direct classification we include multinomial logistic regression and random forests [12, 22, 3, 14]. For neural sequence models we consider recurrent architectures [8] and Transformer-style encoders [21, 27], which have demonstrated strong performance on a range of time series tasks and have motivated specialized variants for long-range temporal dependencies [13, 28, 25]. Features are constructed from standard daily market variables (open, high, low, close, volume) and widely used technical indicators such as RSI, moving averages, MACD, Bollinger Bands, and stochastic oscillators [2, 10, 24, 1]. Price and metadata are obtained from widely used public sources [6, 18, 23].

Across these model families, a central empirical theme emerges: maximizing headline accuracy can encourage *prediction concentration*, where a model frequently predicts only a small subset of stocks that dominate the label distribution. In our experiments, linear and tree-based classifiers can achieve the highest test accuracy in this benchmark, but often by focusing predictions on a few highly volatile assets. In contrast, Transformer-based models typically yield lower top-1 accuracy yet produce a broader distribution over predicted winners, which may be preferable when the classifier is a component inside a larger decision pipeline. This motivates a more nuanced view of evaluation for relative-return prediction tasks, in which diversity-aware summaries complement standard classification metrics.

Contributions. This work makes the following contributions:

- **Decision-aligned formulation.** We formalize short-horizon equity selection as a *winner prediction* problem, reframing relative-return forecasting as a multiclass classification task that aligns naturally with sequence modeling and next-token prediction objectives.
- **Unified empirical comparison.** We conduct a controlled comparison between forecast-then-argmax pipelines and direct discriminative models, spanning classical time-series methods, standard classifiers, and neural sequence models, under a consistent temporal split and feature construction.
- **Evaluation failure mode.** We empirically demonstrate that optimizing top-1 accuracy under label imbalance can induce *prediction concentration*, where models collapse onto a small subset of dominant assets, obscuring poor class coverage.
- **Accuracy–concentration tradeoff.** We show that different model families occupy distinct points on a tradeoff between predictive accuracy and diversity of predicted winners, with Transformer encoders favoring broader class coverage at the cost of accuracy.
- **Implications for Financial AI evaluation.** We argue that relative-performance forecasting tasks require class-sensitive and concentration-aware metrics in addition to accuracy, particularly in small-universe, non-stationary settings.

The remainder of the paper is organized as follows. Section 2 defines the task and evaluation protocol. Section 3 describes the dataset and feature construction. Section 4 presents the models, and Section 5 reports empirical results and analyses. Section 6 discusses implications, limitations, and responsible use.

2 Problem Formulation and Evaluation

Let $\mathcal{A} = \{1, \ldots, N\}$ denote a fixed universe of N stocks. For each asset $i \in \mathcal{A}$ and trading day t, we observe a feature vector $\mathbf{x}_t^{(i)} \in \mathbb{R}^d$ built from daily market variables (e.g., OHLCV), calendar encodings, and technical indicators.

We define the (signed) one-day return for asset i on day t as

$$r_t^{(i)} = 100 \cdot \frac{C_t^{(i)} - O_t^{(i)}}{O_t^{(i)}}, \tag{1}$$

where $O_t^{(i)}$ and $C_t^{(i)}$ are the open and close prices of asset i on day t.

2.1 Winner label (multiclass target)

For each day t, we assign a *winner* label as the index of the asset with the largest realized return within the universe:

$$y_t = \arg\max_{i \in \mathcal{A}} \; r_t^{(i)} \in \{1, \ldots, N\}. \tag{2}$$

Ties are rare; if they occur, they can be broken deterministically (e.g., by choosing the smallest index) to keep the mapping well-defined.

This produces a supervised learning dataset of input histories and next-day labels $\{(\mathbf{X}_t, y_{t+1})\}$, where $\mathbf{X}_t$ summarizes information available up to day t.

We use a fixed-length lookback window of L trading days. Let

$$\mathbf{X}_t = \left[\mathbf{x}_{t-L+1}^{(1)}, \ldots, \mathbf{x}_t^{(1)};\; \ldots;\; \mathbf{x}_{t-L+1}^{(N)}, \ldots, \mathbf{x}_t^{(N)}\right] \tag{3}$$

denote the windowed inputs for all assets. Depending on the model family, $\mathbf{X}_t$ is treated either as a sequence (for LSTM/Transformer encoders) or as a flattened feature vector (for classical classifiers). All splits are chronological to avoid look-ahead bias.

2.2 Two modeling paradigms

We compare two common ways to produce a winner prediction $\hat{y}_{t+1}$.

(i) Continuous forecasting pipeline. A forecaster produces per-asset return predictions $\hat{r}_{t+1}^{(i)}$ and the predicted winner is selected by an argmax:

$$\hat{y}_{t+1} = \arg\max_{i \in \mathcal{A}} \; \hat{r}_{t+1}^{(i)}. \tag{4}$$

This paradigm includes univariate models fit per asset and multivariate models that jointly model the return vector $\mathbf{r}_t = [r_t^{(1)}, \ldots, r_t^{(N)}]^\top$.

(ii) Direct multiclass classification. A classifier directly models the conditional distribution over winners:

$$\hat{\mathbf{p}}_{t+1} = f_\theta(\mathbf{X}_t) \in \Delta^{N-1}, \qquad \hat{y}_{t+1} = \arg\max_{k \in \{1,\ldots,N\}} \; \hat{p}_{t+1,k}, \tag{5}$$

where Δ^{N-1} is the $(N-1)$-simplex. This formulation mirrors next-token prediction in sequence modeling, with classes corresponding to assets.

3 Data and Features

Data source and stock universe. We study a fixed universe of $N = 7$ large-cap technology stocks drawn from the NASDAQ-100 constituents with the highest market capitalization at the time of data collection: Apple (AAPL), Amazon (AMZN), Alphabet Class C (GOOG), Meta (META), Microsoft (MSFT), NVIDIA (NVDA), and Tesla (TSLA). The motivation for focusing on this small, liquid universe is twofold: (i) these assets are heavily traded and have relatively stable data availability, and

(ii) the constrained setting enables a controlled comparison of modeling paradigms under a consistent evaluation protocol.

Daily OHLCV data (Open, High, Low, Close/Last, Volume) and trading dates are obtained from the official NASDAQ data service [18]. The curated datasets used in this work are provided as supplementary materials. The raw series covers the maximum historical window available from the NASDAQ interface at the time of extraction, with the earliest date on May 7, 2015 and the latest date on May 6, 2025 (2,515 daily records per stock prior to feature engineering) [18].

Preprocessing and cleaning. We use daily OHLCV data for each stock and define the target return as the open-to-close percentage change $r_t^{(i)} = 100 \cdot (C_t^{(i)} - O_t^{(i)})/O_t^{(i)}$ (Eq. (1)), labeling each day by the stock with the maximum next-day return (Eq. (2)). Features include the base price/volume variables, cyclical calendar encodings (weekday, month, week-of-year), and standard technical indicators (RSI and ATR [24], Bollinger Bands [2], stochastic oscillator [10], and common moving-average and MACD variants), consistent with prior work [1, 7]. Preprocessing is fit on the training period only and applied to validation/test to avoid leakage; rolling-window indicators induce initial undefined values, so we drop the first 19 rows per stock after feature engineering, retaining outliers as genuine market events and standardizing continuous inputs with `StandardScaler`. The final dataset contains 17,472 rows (2,496 per stock) with 23 columns per row, and we evaluate using a chronological split: train (May 2015–Dec 2022), validation (Jan 2023–Jun 2024), and test (Jul 2024–May 2025). See App. A for additional details.

4 Methods

We study the winner-prediction task (Section 2) through two complementary paradigms. The first is a two-stage *continuous forecasting* approach: forecast the next-day return for each asset and select the predicted winner by an $\arg\max$ rule (Eq. (4)). The second is *direct multiclass classification*: predict the next-day winner label end-to-end (Eq. (5)). All models are trained and selected using the chronological train/validation split described in Section 3.

Continuous forecasting pipelines. In the continuous setting, a model produces one-step-ahead forecasts $\{\hat{r}_{t+1}^{(i)}\}_{i=1}^N$ for the return of each stock, and the winner prediction is obtained as $\hat{y}_{t+1} = \arg\max_i \hat{r}_{t+1}^{(i)}$ (Eq. (4)). We evaluate classical time-series baselines that differ in how they exploit temporal structure and cross-asset dependence. *ARIMA* is fit independently to each asset return series $\{r_t^{(i)}\}$ and combines autoregressive, differencing, and moving-average components [16, 26]:

$$\phi(B)(1-B)^d r_t^{(i)} = \theta(B)\varepsilon_t, \tag{6}$$

where B is the lag operator and $\phi(B)$ and $\theta(B)$ are polynomials of orders p and q. To incorporate correlations across assets, we also consider *VARIMA*, which models the joint return vector $\mathbf{r}_t = [r_t^{(1)}, \ldots, r_t^{(N)}]^\top$ through matrix-valued AR and MA polynomials [16, 26]:

$$\Phi(B)(1-B)^d \mathbf{r}_t = \Theta(B)\boldsymbol{\varepsilon}_t. \tag{7}$$

Finally, to include engineered signals (Section 3), we evaluate *VARMAX*, which augments the multivariate return dynamics with exogenous covariates $\mathbf{u}_t$ (e.g., technical indicators and temporal encodings):

$$\mathbf{r}_t = \sum_{j=1}^{p} \Phi_j \mathbf{r}_{t-j} + \sum_{k=0}^{s} \beta_k \mathbf{u}_{t-k} + \boldsymbol{\varepsilon}_t. \tag{8}$$

For all continuous models, forecasts are converted into winner predictions using the same $\arg\max$ rule, enabling a direct comparison with the classification-based methods.

Direct multiclass classifiers. In the direct setting, the model predicts the winner label without explicitly forecasting per-asset returns. Inputs are derived from the lookback window $\mathbf{X}_t$ (Section 2) and represented as either a fixed-length vector (for standard classifiers) or a sequence (for neural models). We include two classical multiclass baselines. *Multinomial Logistic Regression* models the conditional distribution over winners with a linear score followed by a softmax [12, 3, 14]:

$$\ell_k = \mathbf{w}_k^\top \mathbf{z}_t + b_k, \qquad p(y_{t+1} = k \mid \mathbf{z}_t) = \frac{\exp(\ell_k)}{\sum_{j=1}^N \exp(\ell_j)}, \tag{9}$$

and is trained by minimizing cross-entropy:

$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{|\mathcal{T}_{\mathrm{train}}|} \sum_{t \in \mathcal{T}_{\mathrm{train}}} \log p(y_{t+1} \mid \mathbf{z}_t). \tag{10}$$

Random Forest constructs an ensemble of decision trees with feature subsampling and aggregates predictions by majority vote [22, 3, 14]:

$$\hat{y}_{t+1} = \mathrm{mode}\{h_1(\mathbf{z}_t), \ldots, h_M(\mathbf{z}_t)\}. \tag{11}$$

To preserve the temporal protocol, training is performed on chronologically ordered samples (no time shuffling), while stochasticity is introduced through the ensemble construction.

Another type of multiclass classifiers are neural sequence models that operate directly on the length-L lookback window as a sequence and learn temporal representations that are mapped to a winner distribution. We consider *LSTM* networks that employ gated recurrent updates to capture temporal dependencies while mitigating vanishing gradients [8, 20, 9, 19]. Given the sequence input, the final hidden representation is passed to a linear classifier and trained with cross-entropy; this setup is commonly used in financial prediction with technical indicators [1].

Transformers. *Transformer* encoders replace recurrence in LSTMs with self-attention, enabling parallel processing and explicit modeling of interactions across the entire lookback window [21, 17, 27]. Scaled dot-product attention is defined as

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V, \tag{12}$$

and we use an encoder-only architecture composed of stacked multi-head attention and feed-forward blocks with residual connections and layer normalization [21]. Temporal order is provided via positional encodings or learned positional embeddings, and the encoder output is pooled (or flattened, depending on configuration) and passed to a classification head producing N logits. This formulation aligns naturally with sequence-to-label modeling for time series [27] and is consistent with Transformer variants proposed for long-range forecasting [13, 28, 25]. Hyperparameter choices follow the ranges summarized in [7, 11].

Model selection and reporting. For each method, hyperparameters are selected on the validation period using the metrics in Section 2. The final configuration for each model family is then evaluated on the held-out test period once. We report accuracy and weighted F1, and we additionally summarize the concentration of predicted classes to distinguish models that achieve accuracy by predicting a small subset of winners from those that distribute predictions more broadly.

5 Results

We report test-set performance for all model families under the evaluation protocol in Section 2. Table 1 summarizes the best-performing configuration for each method, selected by validation accuracy, and lists per-class F1-scores together with overall accuracy and the support-weighted F1 (Avg Weight).

Overall comparison. Table 1 shows that the best test accuracy on the 7-stock universe is achieved by multinomial Logistic Regression with all features (27.83%). This is only marginally above a naive majority-class baseline that always predicts the most frequent winner in the test set (TSLA), which wins 26.29% of the time. The Random Forest achieves a similar accuracy (26.42%) but exhibits the same qualitative behavior: strong performance is driven primarily by correctly predicting highly frequent winners.

Within the continuous-forecasting pipeline (predict next-day returns per asset and select $\arg\max$), the best results are obtained by VARIMA using only daily returns (20.19% accuracy), while ARIMA reaches 15.96% and VARMAX (with exogenous features) reaches 16.04% in the best setting. Overall, the continuous models underperform direct classifiers in this benchmark.

Class imbalance and prediction concentration. A key observation from Table 1 is that high accuracy can coincide with poor coverage of minority classes. For LR and RF, per-class F1-scores are zero for most stocks, with performance concentrated on a small subset of frequent winners (notably

Table 1: Model comparison by variable type and method

Type	Model	F1-score by Stock							Acc.	Avg.
		AAPL	AMZN	GOOG	META	NVDA	TSLA	MSFT		
Only Daily Return										
Classification	LR	0.00	0.00	0.00	0.00	0.32	0.39	0.00	26.29	0.17
	RF	0.00	0.00	0.00	0.00	0.31	0.41	0.00	27.23	0.17
Continuous	ARIMA	0.06	0.00	0.08	0.10	0.28	0.18	0.18	15.96	0.15
	VARIMA	0.13	0.11	0.00	0.14	0.28	0.31	0.00	20.19	0.19
All Features										
Classification	LR	0.11	0.00	0.00	0.00	0.37	0.38	0.00	27.83	0.20
	RF	0.12	0.00	0.00	0.00	0.38	0.28	0.00	26.42	0.18
	LSTM	0.34	0.15	0.15	0.00	0.31	0.05	0.00	21.14	0.17
Continuous	VARMAX	0.30	0.00	0.00	0.00	0.04	0.00	0.00	16.04	0.06
Transformer	Transformer	0.08	0.00	0.05	0.01	0.28	0.36	0.03	23.21	0.17

TSLA and NVDA). This indicates that these models often collapse to predicting only a few classes, which is consistent with strong label imbalance in winner prediction.

In contrast, the Transformer encoder attains a lower top-1 accuracy (23.21%) but yields non-zero F1-scores for a broader set of stocks (including smaller classes such as GOOG, META, and MSFT). Among the evaluated methods, it offers the most balanced tradeoff between top-1 accuracy and class-wise coverage. Notably, MSFT obtains non-zero F1 only for ARIMA and the Transformer, while remaining at 0.0000 for most other approaches in the all-features setting.

Effect of removing MSFT. The consistently low F1 for MSFT across most models motivated an ablation in which MSFT is removed from the universe and the full study is repeated on the remaining 6 stocks. Table 2 reports the results of this setting (captioned "excluding MSFT" in the source thesis table). In this modified universe, performance improves for several models, with the most pronounced change for the Transformer: its test accuracy increases from 23.21% to 25.78%, and its weighted F1 increases from 0.1717 to 0.2284. This indicates that universe composition can substantially affect both overall performance and class-wise behavior in winner prediction, and that reducing redundancy among assets can make the next-winner signal easier to learn.

Additional analyses. To further interpret the concentration effects, we also report results on two restricted universes defined by winner frequency: a "big players" subset (AAPL, NVDA, TSLA) and a "small players" subset (AMZN, GOOG, META, MSFT). Results are summariezed in Table 3 and Table 4. In these settings, accuracies increase substantially relative to the full 7-way task, reflecting the reduced number of classes and more homogeneous competition within each subset. Finally, we report performance on a restricted time period (2022–2024) to probe sensitivity to recent market regimes; results remain consistent with the main findings, with LR/RF maintaining competitive accuracy but limited class coverage, and the Transformer trading some accuracy for broader predicted-class support.

Overall, the results support two conclusions: (i) winner prediction is challenging, with many methods failing to outperform a simple majority baseline by a wide margin, and (ii) accuracy alone can obscure meaningful differences in class-wise behavior, making weighted F1 and predicted-class coverage important complementary summaries for this task.

6 Discussion

This study evaluates a decision-aligned formulation of short-horizon equity forecasting: predicting which stock within a fixed universe will achieve the highest next-day return. Recasting the problem as multiclass classification simplifies comparison across model families and connects naturally to sequence modeling objectives used in modern deep learning [21, 27]. At the same time, the formulation exposes practical challenges that are often less visible in regression-style return prediction, most notably label imbalance and prediction concentration.

Table 2: Model comparison by variable type and method excluding MSFT

Type	**Model**	**F1-score by Stock**						**Acc.**	**Avg.**
		AAPL	AMZN	GOOG	META	NVDA	TSLA		
Only Daily Return									
Classification	LR	0.00	0.00	0.00	0.00	0.20	0.35	22.07	0.14
	RF	0.00	0.00	0.00	0.00	0.25	0.39	25.82	0.16
Continuous	ARIMA	0.08	0.00	0.17	0.20	0.28	0.17	18.31	0.16
	VARIMA	0.26	0.00	0.05	0.15	0.35	0.18	22.54	0.20
All Features									
Classification	LR	0.13	0.00	0.00	0.00	0.36	0.38	27.83	0.20
	RF	0.00	0.00	0.00	0.00	0.37	0.24	24.06	0.14
	LSTM	0.05	0.06	0.21	0.12	0.30	0.08	16.34	0.14
Continuous	VARMAX	0.00	0.00	0.00	0.00	0.00	0.41	25.94	0.11
Transformer	Transformer	0.23	0.02	0.10	0.11	0.30	0.34	25.78	0.23

Table 3: Model comparison of big players

Type	**Model**	**F1-score by Stock**			**Acc.**	**Avg.**
		AAPL	NVDA	TSLA		
Only Daily Return						
Classification	LR	0.33	0.32	0.40	35.21	0.35
	RF	0.11	0.40	0.33	31.92	0.27
Continuous	ARIMA	0.35	0.39	0.22	32.86	0.32
	VARIMA	0.29	0.34	0.38	33.80	0.33
All Features						
Classification	LR	0.22	0.17	0.47	33.96	0.29
	RF	0.00	0.48	0.10	31.60	0.18
	LSTM	0.34	0.22	0.38	33.00	0.32
Continuous	VARMAX	0.03	0.00	0.50	33.49	0.17
Transformer	Transformer	0.00	0.00	0.52	35.25	0.18

Table 4: Model comparison of small players

Type	**Model**	**F1-score by Stock**				**Acc.**	**Avg.**
		AMZN	GOOG	META	MSFT		
Only Daily Return							
Classification	LR	0.32	0.26	0.34	0.28	30.05	0.30
	RF	0.29	0.35	0.30	0.14	29.11	0.28
Continuous	ARIMA	0.03	0.31	0.27	0.19	23.00	0.20
	VARIMA	0.00	0.00	0.40	0.00	25.35	0.10
All Features							
Classification	LR	0.25	0.38	0.20	0.16	28.30	0.25
	RF	0.27	0.20	0.35	0.04	26.89	0.23
	LSTM	0.16	0.41	0.13	0.12	25.48	0.21
Continuous	VARMAX	0.00	0.03	0.04	0.34	21.23	0.09
Transformer	Transformer	0.00	0.42	0.00	0.00	26.23	0.11

Accuracy versus prediction concentration A recurring pattern across our experiments is that strong headline accuracy can coincide with highly concentrated predictions. Linear and tree-based classifiers, especially when supplied with richer technical features, can improve test accuracy but often do so by predicting a small subset of stocks that dominate the winner distribution. In our setting these are typically the most volatile assets, for which the probability of producing the maximum return on any given day is higher. This behavior is consistent with optimizing standard cross-entropy or impurity-based objectives under class imbalance: predicting a frequent class is rewarded more often, and the model may learn decision boundaries that effectively ignore minority winners.

Advantage of Transformers. In contrast, Transformer encoders tend to produce a broader distribution of predicted winners, even when their top-1 accuracy is lower. This does not imply that the Transformer is intrinsically superior for this task, but it suggests that sequence models may capture different temporal regularities and class interactions that discourage collapse to a small set of labels. From a Financial AI perspective, this difference matters because winner prediction is rarely an end in itself; it is commonly a component inside a larger decision pipeline (e.g., allocation, risk constraints, or an agentic trading system). In such settings, a model that always recommends the same few assets can be brittle under regime changes and may be undesirable for diversification or risk management, even if its accuracy appears competitive in a static evaluation.

Why classical time-series models underperform. ARIMA and multivariate extensions provide transparent baselines grounded in classical time-series analysis [16, 26]. However, their relative underperformance here is unsurprising. First, one-day equity returns are noisy and have weak linear autocorrelation; second, the winner label depends on the *relative* cross-asset maxima rather than the absolute predictability of any single series. In addition, the mapping from continuous return forecasts to a discrete winner via an $\arg\max$ is sensitive to small forecasting errors: even when per-asset forecasts are reasonable, the induced winner ranking can be unstable. This highlights a key benefit of direct classification, which learns the decision boundary on the induced label space rather than relying on a two-step pipeline.

Feature sensitivity and universe dependence Feature engineering with common technical indicators can improve predictive performance for several model families, but it also amplifies the concentration effect for simpler classifiers. This suggests that technical indicators may increase separability for the dominant classes while remaining insufficient to disambiguate minority winners. Additionally, the observed improvement in Transformer performance under a fixed-universe ablation (removing MSFT) underscores that the winner task is sensitive to the universe composition. Because the label is defined by a cross-sectional maximum, adding or removing a single asset can change the class distribution and the difficulty of the decision boundary. This has implications for benchmarking: results should be interpreted as properties of the *task instance* (universe, horizon, period), not as universal statements about model superiority.

7 Conclusion

We investigated next-day *winner prediction* for a small universe of large-cap equities by framing relative outperformance as a multiclass classification problem. Across experiments, we find that while simple classifiers can achieve the highest top-1 accuracy, they often do so by concentrating predictions on a small subset of dominant classes. Transformer encoders, although typically less accurate, produce a broader distribution of predicted winners, suggesting a different tradeoff that may be preferable in downstream decision systems where robustness and diversification matter.

References

[1] Hum Nath Bhandari, Binod Rimal, Nawa Raj Pokhrel, Ramchandra Rimal, Keshab Dahal, and Rajendra K C Khatri. Predicting stock market index using lstm. *Machine Learning with Applications*, 9:100320, 05 2022. doi: 10.1016/j.mlwa.2022.100320.

[2] John Bollinger. *Bollinger on Bollinger Bands*. McGraw-Hill, New York, 2001.

[3] Giovanni Campisi, Silvia Muzzioli, and Bernard De Baets. A comparison of machine learning methods for predicting the direction of the us stock market on the basis of volatility indices. *International Journal of Forecasting*, 40(3):869–880, 2024. ISSN 0169-2070. doi: https://doi.

org/10.1016/j.ijforecast.2023.07.002. URL https://www.sciencedirect.com/science/article/pii/S0169207023000729.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

[5] Eugene F. Fama. Random walks in stock market prices. *Financial Analysts Journal*, 51(1): 75–80, 1995. ISSN 0015198X. URL http://www.jstor.org/stable/4479810.

[6] Yahoo Finance. Yahoo finance, 2025. URL https://finance.yahoo.com.

[7] Tizian Fischer, Marius Sterling, and Stefan Lessmann. Fx-spot predictions with state-of-the-art transformer and time embeddings. *Expert Systems with Applications*, 249:123538, 2024. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2024.123538. URL https://www.sciencedirect.com/science/article/pii/S0957417424004032.

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[9] Dariusz Kobiela, Dawid Krefta, Weronika Król, and Paweł Weichbroth. Arima vs lstm on nasdaq stock exchange data. *Procedia Computer Science*, 207:3830–3839, 10 2022. doi: 10.1016/j.procs.2022.09.445.

[10] George C. Lane. The stochastic indicator. *Stocks & Commodities Magazine*, 4(1):50–56, 1986.

[11] Shuozhe Li, Zachery B Schulwolf, and Risto Miikkulainen. Transformer based time-series forecasting for stock. *arXiv:2502.09625*, January 2025. URL http://www.cs.utexas.edu/users/ai-lab?li:arxiv25.

[12] Syed Shahan Li, Muhammad Mubeen, and Adnan Hussain. Prediction of stock performance by using logistic regression model: Evidence from pakistan stock exchange (psx). *Asian Journal of Empirical Research*, 15:212, 2018.

[13] Bryan Lim, Sercan O. Arik, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting, 2020. URL https://arxiv.org/abs/1912.09363.

[14] Yuang Liu. Stock prediction analysis based on logisticregression and random forest. *MedScien*, 1(3), 2025.

[15] Burton G. Malkiel. *Efficient Market Hypothesis*. Palgrave Macmillan UK, London, 1989. ISBN 978-1-349-20213-3. doi: 10.1007/978-1-349-20213-3_13. URL https://doi.org/10.1007/978-1-349-20213-3_13.

[16] Prapanna Mondal, Labani Shit, and Saptarsi Goswami. Study of effectiveness of time series modeling (arima) in forecasting stock prices. *International Journal of Computer Science, Engineering and Applications*, 4:13–29, 04 2014. doi: 10.5121/ijcsea.2014.4202.

[17] Leila Mozaffari and Jianhua Zhang. Predictive modeling of stock prices using transformer model. In *Proceedings of the 2024 9th International Conference on Machine Learning Technologies*, ICMLT '24, pp. 41–48, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400716379. doi: 10.1145/3674029.3674037. URL https://doi.org/10.1145/3674029.3674037.

[18] Nasdaq. Market activity - quotes, 2025. URL https://www.nasdaq.com/market-activity/stocks.

[19] Sima Siami-Namini and Akbar Siami Namin. Forecasting economics and financial time series: Arima vs. lstm, 2018. URL https://arxiv.org/abs/1803.06386.

[20] Sima Siami-Namini and Akbar Siami Namin. Forecasting economics and financial time series: Arima vs. lstm, 2018. URL https://arxiv.org/abs/1803.06386.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[22] Mehar Vijh, Deeksha Chandola, Vinay Anand Tikkiwal, and Arun Kumar. Stock closing price prediction using machine learning techniques. *Procedia Computer Science*, 167:599–606, 2020. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2020.03.326. URL https://www.sciencedirect.com/science/article/pii/S1877050920307924. International Conference on Computational Intelligence and Data Science.

[23] Wikipedia contributors. Anexo:empresas por capitalización de mercado — wikipedia, la enciclopedia libre, 2024. URL https://es.m.wikipedia.org/wiki/Anexo%3AEmpresas_por_capitalizaci%C3%B3n_de_mercado. Consultado el 3 de julio de 2025.

[24] J. Welles Jr. Wilder. *New Concepts in Technical Trading Systems*. Trend Research, New York, 1978.

[25] Qin Wu, Zhaocheng Xu, Haixu Wang, Ailing Zeng, Qiang Xu, and Qi Tian. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:22419–22430, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/6049d9c3d56fa6079677084525e0eac9-Abstract.html.

[26] Yuchen Xiang. Forecasting the nasdaq index based on the arima model. *Advances in Economics, Management and Political Sciences*, 141:206–213, 12 2024. doi: 10.54254/2754-1169/2024.GA18869.

[27] George Zerveas, Sercan Jayaraman, Dhaval Patel, Abhinav Bhamidipaty, and Carsten Eickhoff. A transformer-based framework for multivariate time series representation learning. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2021.

[28] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11106–11115, 2021. doi: 10.1609/aaai.v35i12.17325. URL https://ojs.aaai.org/index.php/AAAI/article/view/17325.

A Additional Preprocessing and cleaning details

All splits are chronological (Section 2) and preprocessing steps are fit on the training portion only, then applied to validation and test to avoid information leakage.

Missing values and indicator warm-up. The raw NASDAQ OHLCV series contains no missing values. However, technical indicators require rolling windows (e.g., moving averages and Bollinger Bands), which produce undefined values at the beginning of each stock's history. After feature engineering, the first 19 rows per stock contain at least one null value (driven by Bollinger Band computation) and are removed. This yields a final dataset of 17,472 rows (2,496 observations per stock), spanning June 4, 2015 to May 6, 2025.

Outliers. Large daily moves, particularly for TSLA and NVDA, are retained as they reflect genuine market events and strongly influence the winner label distribution. To reduce scale sensitivity during training, we standardize continuous inputs as described below.

Scaling. Continuous features are standardized using `StandardScaler` (zero mean, unit variance). This choice is preferable to min-max scaling in the presence of large outliers, and it improves numerical stability for scale-sensitive models.

A.1 Base variables and target definition

For each stock i and day t, the base observed variables are:

$$\{\texttt{Open}, \texttt{High}, \texttt{Low}, \texttt{Close/Last}, \texttt{Volume}, \texttt{Date}\}.$$

The target return used to define winners is the daily open-to-close percentage return:

$$r_t^{(i)} = 100 \cdot \frac{C_t^{(i)} - O_t^{(i)}}{O_t^{(i)}},$$

consistent with Eq. (1). Each day is labeled with the index of the stock achieving the maximum return within the universe (Eq. (2)).

A.2 Feature engineering

In addition to the base variables, we construct two groups of features: temporal encodings and technical indicators.

Temporal variables (cyclical encoding). We derive three calendar features: day of week, month, and week number of year. Because these variables are cyclical, we encode each using sine/cosine transformations to preserve circular continuity (e.g., December adjacent to January). This avoids the artificial discontinuities introduced by one-hot encoding for periodic categories.

Technical indicators. To enrich short-horizon signals and support deep sequence models, we compute widely used indicators, following standard definitions: (i) Relative Strength Index (RSI) [24], (ii) Simple Moving Average (SMA), (iii) Exponential Moving Average (EMA), including EMA_{12} and EMA_{26}, (iv) Moving Average Convergence Divergence (MACD), with signal and histogram components, (v) Bollinger Bands [2], (vi) Average True Range (ATR) [24], and (vii) Stochastic Oscillator (%K) [10]. This feature expansion mirrors common practice in prior forecasting studies using neural sequence models [1, 7].

Final feature set. After concatenation across stocks, an identifier column `Stock` is retained to track the asset associated with each observation (while the winner label remains the supervised target). The final dataset contains 23 columns per row, consisting of OHLCV variables, engineered temporal features, and technical indicators.

A.3 Temporal split

We adopt a forward-looking split aligned to calendar boundaries:

- **Training:** May 2015 through December 2022 ($\approx 75\%$).
- **Validation:** January 2023 through June 2024 ($\approx 15\%$).
- **Test:** July 2024 through early May 2025 ($\approx 10\%$).

This split supports realistic evaluation under distribution shift and avoids look-ahead bias. After selecting configurations using validation performance, the final reported models are retrained on the combined training+validation set and evaluated once on the held-out test period.