Louvain School of Management

The Impact of Prediction Market Signals on Stock Prices: A Comparative Exploratory Study of Global vs. Local Modeling Techniques

Author(s): Adrien Payen
Supervisor(s): Corentin Vande Kerckhove
Academic year 2024-2025
Dissertation for the master of
Business Analytics
Daytime schedule

Abstract

Acknowledgments

I express my sincere gratitude to Delphia for the trust they have placed in me throughout this project. Their support has been invaluable and I am especially grateful for the data they provided which formed the backbone of my research.

I would also like to extend my heartfelt thanks to Professor Vande Kerckhove for his guidance, feedback and constant encouragement. His expertise has been instrumental in shaping this research and helping me navigate the complexities of this topic.

Finally, I would like to acknowledge all those who have contributed to this project, either directly or indirectly. Your support, insight and encouragement have been greatly appreciated. I am grateful to each one of you.

Contents

1 Introduction

In a rapidly evolving financial landscape, the role of information in shaping market behavior has become increasingly significant. One of the most intriguing and complex challenges is understanding the relationship between prediction markets and their influence on stock returns movements. This thesis addresses the problem of quantifying how predictive signals derived from prediction markets impact stock returns.

1.1 The Current World of Prediction Markets and Stock Prices

In today's financial world, markets are flooded with a wide range of information that can impact investor behavior (Li et al., 2014). Prediction markets, also known as event futures or information markets, aggregate individual opinions and probabilities about future events, offering valuable signals about the direction of various economic, political or social factors (Waitz and Mild, 2013; Wolfers and Zitzewitz, 2004). These markets provide a unique opportunity for investors to gauge future expectations and anticipate market movements.

1.2 Research Question

This thesis focuses on the research question: *How do prediction markets signals affect stock returns?* To answer this, we explore the potential mechanisms through which these signals influence investor behavior and, by extension, the market valuation of assets. By investigating prediction markets signals, this research aims to build a comprehensive model for stock price prediction that accounts for the full spectrum of information available to investors.

To address this question, we adopt a comparative modeling strategy. Specifically, we assess the relevance and performance of a global modeling approach — a single regression model applied across all stocks, incorporating both returns and exogenous variables common to all assets — versus a stock-specific model, where a distinct model is trained for each individual stock. This methodology enables to determine whether a shared structure, enriched by exogenous information from prediction markets, can generalize across different stocks, or whether asset-specific nuances require tailored models.

1.3 Managerial Relevance

This research holds substantial managerial relevance for Delphia, a Canadian fintech specializing in data-driven investment strategies. The implications are outlined below :

1. Enhanced data-driven investment strategies: By analyzing how predictive signals from prediction markets influence stock prices, Delphia can refine its investment models. Integrating these signals into algorithms could enhance decision-making processes and potentially improve portfolio performance (Waitz and Mild, 2013).

2. Improved forecasting accuracy: Prediction markets aggregate public sentiment and probabilities about future events, offering valuable insights to anticipate market trends. By combining these signals with Delphia's existing data models, the accuracy of stock price predictions could be significantly improved, leading to more informed investment decisions and better risk management.

3. Detection of market inefficiencies and early trends: This research can aid Delphia in identifying market inefficiencies and emerging trends by leveraging prediction market data. This proactive approach could allow the company to capitalize on shifts in market dynamics before competitors.

4. Risk management optimization: Incorporating prediction market signals would bolster Delphia's ability to detect potential risks during periods of volatility or uncertainty. This enhanced risk management capability could help protect the company's portfolios from significant losses and reduce exposure to high-risk investments.

5. Innovation in investment products: The findings could inspire the development of new financial products that utilize prediction market signals. For instance, funds tailored to exploit trends from these markets could offer clients innovative, high-return investment options.

6. Competitive advantage through strategic information use: Expanding Delphia's data integration to include prediction markets could provide a distinct competitive edge. By harnessing diverse information sources, the company can position itself as a leader in identifying and acting on emerging trends, attracting clients and increasing its market presence.

Overall, this research aligns closely with Delphia's mission of leveraging data to inform investment decisions. The insights gained can optimize strategies, improve predictive accuracy and strengthen the company's position in the fintech sector.

1.4 Thesis Structure

This thesis is organized into chapters. The second chapter, Literature Review, examines the existing literature on prediction markets, stock price dynamics and recent advancements in global modeling applied to finance. This sets a comprehensive foundation for the research. In the third chapter, Methodology, the focus is on data collection methods, the techniques employed to implement global modeling and the integration of prediction markets to enhance the understanding of stock market movements. This chapter provides a detailed explanation of the methodological framework, including the implementation of prediction market data and global modeling. Chapter four, Analysis of the results, presents the findings of the empirical analysis, showcasing how prediction markets influence stock prices and evaluating the effectiveness of global modeling techniques in interpreting these dynamics. The fifth chapter, Discussion, interprets the results, compares them with existing literature and discusses their implications for both theory and practice. Finally, the Conclusion summarizes the research, highlights its implications for investors and regulators and suggests areas for future research.

2 Literature Review

This section examines previous theories and research on the valuation of information and its influence on financial markets. It reviews key studies, theoretical frameworks and findings relevant to understanding the connection between prediction markets and stock market responses.

2.1 Prediction Markets

Prediction markets are platforms where participants trade contracts that pay out based on the outcome of future events (Wolfers and Zitzewitz, 2004). These markets aggregate the individual predictions of participants, turning them into a collective forecast of the likelihood that an event will occur. The concept of prediction markets is rooted in the idea that the collective wisdom of a diverse group of individuals, each with access to different information and insights, can generate more accurate forecasts than any individual expert (Bossaerts et al., 2022). These markets function similarly to financial markets (Wolfers and Zitzewitz, 2004), where participants buy and sell contracts based on their predictions of future outcomes. For instance, participants might buy a contract that pays out if a specific political event occurs, such as the election of Trump against Kamala Harris during the 2024 presidential race (Mongrain and Stegmaier, 2024), or if a company meets a certain financial target.

One of the key features of prediction markets is their ability to aggregate information. As discussed by Bossaerts et al., 2022, these markets incorporate diverse viewpoints, enabling them to reflect the collective intelligence of participants. This phenomenon is often referred to as the ”wisdom of crowds”. Berg et al., 2008, have shown that prediction markets can outperform individual experts in terms of forecasting accuracy, as they incorporate a wide range of opinions.

These markets are typically structured as winner-take-all markets or vote-share markets. In winner-take-all markets (Dai et al., 2021), participants predict a single outcome and those who predict the correct event share the prize. This market model offers a fixed payout for the correct prediction, which is typically the same for all winners. Vote-share markets involve trading based on the estimated proportion of votes or shares in a specific outcome, often used for more detailed or continuous predictions, such as estimating the exact vote share in an election or market share in a competition (Dai et al., 2021).

Research has shown that prediction markets can effectively predict not only political events but also economic trends, such as the likelihood of a recession or the movement of stock prices (Wolfers and Zitzewitz, 2004). These markets function as a tool to forecast economic indicators, company performance, or even geopolitical events. By integrating real-time information and the collective insights of a diverse set of participants, prediction markets offer a unique and valuable source of predictive signals that can inform decision-making in various domains, including finance. For example, Google estimates its market capitalization using prediction markets to forecast its value prior to an initial public offering (Berg et al., 2009).

2.2 Conceptual and theoretical framework

2.2.1 Efficient Market Hypothesis (EMH)

The Efficient Market Hypothesis (EMH), developed by Eugene Fama, posits that asset prices fully and instantaneously reflect all available information (Naseer, 2016). This theory assumes that markets are highly efficient in processing and integrating information into prices, leaving no opportunity for consistent outperformance based on publicly available data.

In this context, prediction markets can be viewed as a mechanism to enhance market efficiency by aggregating dispersed and heterogeneous information from a diverse set of participants (Downey, 2024). These markets function by incentivizing participants to trade based on their knowledge, insights, or expectations of future outcomes, effectively pooling collective intelligence.

Unlike traditional financial markets, which rely on a combination of fundamental and technical analysis, prediction markets are explicitly designed to reflect probabilities of future events (Nti et al., 2020; Naseer, 2016). By doing so, they may uncover ”hidden” information that would otherwise remain unintegrated into asset prices. This ability to aggregate a wide array of viewpoints and data sources positions prediction markets as a complementary tool for improving the informational efficiency of financial markets.

2.2.2 Investor behavior

Herding : Herding behavior is a significant phenomenon in financial markets, where investors mimic the actions of others rather than relying on their own private information. This tendency often arises during periods of market stress, leading to deviations of asset prices from their intrinsic values. Research highlights that herding is particularly pronounced in emerging markets, where information asymmetry and market inefficiencies are more prevalent (Ah Mand et al., 2023). For example, in the Malaysian stock market, evidence suggests that herding behavior exhibits non-linear characteristics, with variations between up and down market conditions. Shariah-compliant stocks tend to demonstrate herding more significantly during upward market movements, while conventional stocks show limited evidence of herding. This behavior has critical implications for market stability, as it may amplify volatility and hinder market efficiency.

Financial influencers : Financial influencers on social media platforms, particularly those categorized as ”mega influencers” with over one million followers, have a unique ability to shape investor behavior and market dynamics. Studies demonstrate that posts from these influencers can significantly affect investor attention, trading volume and stock price volatility (Keasey et al., 2024). However, their influence on stock returns is limited, requiring posts with extreme sentiment from top influencers to elicit short-lived impacts on returns. This aligns with the ”noise trader” hypothesis, which posits that uninformed trading triggered by such influencers introduces temporary mispricing that reverses over time. By analyzing over 16 million Instagram posts, researchers have highlighted the importance of sentiment and engagement metrics, such as comment volume, in amplifying the visibility and potential market impact of influencer content. These findings underscore the dual-edged role of influencers in promoting market participation while potentially fostering instability through noise trading.

3 Methodological Framework

This section outlines the conceptual and theoretical underpinnings of the research. It begins with a comparative modeling strategy employed, contrasting global versus local models, and explores the theoretical principles, such as the bias-variance tradeoff, guiding this choice. Subsequently, it articulates the testable hypotheses that stem from this framework. The research design is then detailed, covering the overall experimental architecture, the employing temporal validation methods like walk-forward analysis with specific parameters used in the implementation, and the considerations for its practical execution. Finally, it describes the evaluation methodology, including the selection of Root Mean Squared Error (RMSE) as the primary performance metric and the use of the Diebold-Mariano test for comparing forecasting model efficacy, as implemented in the analysis.

3.1 Comparative Modeling Strategy: Global vs. Local Approaches

In financial time series forecasting, one crucial methodological decision involves the level of aggregation in model training — whether to use a single unified model across multiple assets (global modeling) or to estimate distinct models for each asset (local modeling). This distinction has important implications for model complexity, data efficiency, and forecasting accuracy. The present research implements and compares both approaches in the context of predicting daily stock returns using signals from prediction markets as exogenous inputs.

3.1.1 Global Model

A global model is trained using the pooled data from all stocks in the dataset. In this study, a tree-based ensemble algorithm — specifically XGBoost — was used for global modeling due to its ability to handle high-dimensional, heterogeneous features and capture non-linear interactions. The premise of global modeling lies in the assumption that underlying drivers of stock returns share common structures across assets — such as macroeconomic shocks, investor sentiment, or systemic risk factors — that can be learned jointly (Hartford et al., 2018; Gu et al., 2020).

By training a model across all series simultaneously, the global approach benefits from a larger effective sample size, which typically reduces variance and improves generalizability (Montero-

Manso and Hyndman, 2021). Furthermore, the presence of exogenous variables — such as aggregated sentiment from prediction markets — reinforces the potential effectiveness of a global structure by exploiting shared informational signals. However, global models may suffer from specification bias if individual assets exhibit idiosyncratic behaviors that cannot be adequately captured by a single unified function.

XGBoost (Extreme Gradient Boosting) is a powerful machine learning algorithm, introduced by Chen and Guestrin, 2016, used for regression and classification tasks, particularly well-suited for handling large datasets with a mix of numerical and categorical features. It is based on gradient boosting, where multiple weak learners (usually decision trees) are combined to create a strong predictive model.

The mathematical formulation for XGBoost is built upon the idea of boosting an ensemble of trees by minimizing a loss function. Let $F(x)$ represent the prediction for the input x, and each iteration t builds a new tree $h_t(x)$ to minimize the residual error. The model is updated as follows:

$$F_t(x) = F_{t-1}(x) + \eta h_t(x)$$

- $F_t(x)$: prediction at iteration t,
- $F_{t-1}(x)$: prediction from the previous iteration,
- η: learning rate,
- $h_t(x)$: tree added at iteration t.

The loss function $L(\theta)$ used in XGBoost is typically a combination of a regularization term and a loss term (e.g., squared error for regression):

$$L(\theta) = \sum_{i=1}^{N} \ell(y_i, F(x_i)) + \sum_{t=1}^{T} \Omega(h_t)$$

- $\ell(y_i, F(x_i))$: loss function (e.g., mean squared error),
- $\Omega(h_t)$: regularization term to prevent overfitting,
- N: number of observations,

- T: total number of trees.

In this study, XGBoost is used for the global model, leveraging aggregated signals from prediction markets along with stock-specific features to predict stock returns. The model captures complex interactions between various features and allows for the use of high-dimensional data without overfitting.

3.1.2 Local Models

In contrast, local models are estimated separately for each stock. Each asset is modeled independently, allowing the estimation process to capture its unique autoregressive patterns and responsiveness to prediction market signals. For this study, we implemented ARIMA and ARIMAX models — classical time series models introduced by Box et al., 2015 — which are well-suited to univariate or low-dimensional time series data with clear temporal dependencies. The local approach assumes that stocks respond to signals in a firm-specific manner, which may reflect structural differences in sectors, liquidity, or investor composition (Lopez de Prado, 2018; Tashman, 2000).

The general formula for an ARIMA(p,d,q) model is:

$$Y_t = \mu + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \epsilon_t$$

Where:

- Y_t is the observed value at time t,
- μ is the constant mean,
- ϕ_i are the parameters for the autoregressive part (AR),
- ϵ_t is the error term (white noise),
- θ_j are the parameters for the moving average part (MA),
- d represents the order of differencing applied to make the series stationary.

The ARIMA model can also incorporate exogenous variables through the ARIMAX model, which extends ARIMA to include additional independent variables X_t. The ARIMAX model can be written as:

$$Y_t = \mu + \sum_{i=1}^{p} \phi_i Y_{t-i} + \sum_{j=1}^{q} \theta_j \epsilon_{t-j} + \sum_{k=1}^{m} \gamma_k X_{t-k} + \epsilon_t$$

Where :

- X_{t-k} represents the exogenous variables. In this study, the exogenous variables are the prediction market signals, which provide valuable information that might impact stock prices.

Local modeling allows for lower bias by fitting asset-specific dynamics; however, it comes at the cost of higher estimation variance, especially when limited historical data is available per asset. This tradeoff becomes particularly acute in short time series, where overfitting and parameter instability can degrade forecasting performance. From a practical standpoint, maintaining and updating hundreds of separate models also poses operational complexity, especially in high-frequency or live trading environments.

3.1.3 Bias-Variance Tradeoff and Model Evaluation

This dual modeling approach is grounded in the classical bias-variance tradeoff (Geman et al., 1992). Global models, by pooling data, offer stable predictions with lower variance but may introduce systematic bias if heterogeneity across series is substantial. Local models are more flexible and tailored, potentially reducing bias, but are more prone to variance-driven errors. This empirical analysis evaluates these tradeoffs in the context of out-of-sample forecasting error, using Root Mean Squared Error (RMSE) and the Diebold-Mariano test to compare model performance statistically across assets and time periods.

3.2 Testable Hypotheses Derived from the Framework

The following hypotheses guided the empirical investigation:

- **H1:** The informational value of prediction market signals for stock return forecasting is more pronounced in local (stock-specific) models than in global (market-wide) models. This explores whether local models better leverage granular prediction market information.

- **H2:** The influence of prediction market signals differs between individual stocks and broader ETFs. This hypothesis examines whether prediction markets have a more direct impact on firm-specific assets compared to diversified instruments that aggregate multiple companies or sectors.

4 Data and Preprocessing

This section describes the data sources and the preprocessing steps applied to prepare the datasets for empirical modeling, distinguishing between the procedures used for the global and local modeling strategies. The implementation was done in `R`, utilizing a wide range of packages including `tidymodels`, `modeltime`, `timetk`, `tidyverse`, `lubridate`, `KFAS`, `tseries`, `urca`, and others. All analyses and source code are available on the following public GitHub repository, which was made publicly accessible to ensure transparency and reproducibility: https://github.com/ixmxdrien/Master-Thesis.

4.1 Data Sources

The empirical analysis is based on three primary categories of data, each serving a complementary purpose in the modeling and evaluation process:

- **Prediction Market Data (Kalshi):** Forecast data were sourced from Kalshi (https://kalshi.com), a regulated prediction market platform that offers contracts on a broad spectrum of macroeconomic and firm-specific events. The dataset included both quarterly contracts (e.g., Tesla vehicle production, Netflix subscriber growth, Meta daily active users, U.S. GDP growth) and annual contracts (e.g., hurricane counts, SpaceX launches, tech layoffs, Apple car announcements, Ethereum price, measles incidence). Several signals were updated multiple times per day, reflecting real-time changes in market expectations. The data, provided in CSV (e.g., `kalshi-chart-data-metadap-24-q1.csv`), contained time-stamped forecast values or percentages. Custom parsing functions developed in `R` were employed to standardize these inputs and resample them to a daily frequency, making them suitable for integration into time series models.

- **Financial Market Data (Delphia):** Stock return data for a curated selection of U.S. equities were provided by Delphia, a data-driven fintech company. These returns, compiled in the file `tilt_stocks_2024.csv`, include daily observations for prominent stocks such as TSLA, NFLX, META, AAPL, NVDA, MSFT, and others. This dataset served as the primary dependent variable for evaluating the predictive power of Kalshi-based signals on individual company returns.

- **ETF Market Data (Yahoo Finance via `yfinance`):** To investigate whether the relationships observed between Kalshi signals and individual stock returns also extend to broader asset classes, exchange-traded funds (ETFs) were incorporated into the analysis. A Python script leveraging the `yfinance` library was used to query and download returns data for tree major ETFs:

  - S&P 500 (`SPY`)
  - MSCI World Index (`IWDA.AS`)
  - Nasdaq 100 (`QQQ`)

  These ETFs were chosen to represent a range of geographies and sector exposures, allowing the study to explore whether insights derived from Kalshi signals are consistent across both equity-specific and index-based instruments.

4.2 Common Preprocessing Pipeline

For both the global and local models, the following preprocessing steps were systematically applied:

- **Daily Aggregation:** Raw Kalshi prediction market data came with heterogeneous temporal structures. To ensure compatibility with daily financial returns, all prediction signals were aggregated to a daily frequency using a custom function (`process_df_daily`). This function computed the mean of all intraday values for each contract and date, producing one representative prediction value per day per event.

Timestamp	ticker	Value	quarter
2024-01-26 14:00:00	Tesla	481566.5	Q1
2024-01-26 15:00:00	Tesla	481711.8	Q1
2024-01-26 18:00:00	Tesla	480876.7	Q1
2024-01-26 21:00:00	Tesla	480876.7	Q1
2024-01-27 00:00:00	Tesla	480988.4	Q1
2024-01-27 15:00:00	Tesla	484911.4	Q1
2024-01-27 18:00:00	Tesla	485746.9	Q1
2024-01-27 21:00:00	Tesla	485746.9	Q1

date	id	pred daily
2024-01-26	Tesla	481257.9
2024-01-27	Tesla	484348.4

- **Missing Value Imputation:** The handling of missing data differed depending on the data type:
  - *Prediction Market Data:* Between prediction periods (e.g., from Q1 to Q2), some dates were missing due to inactive or unpublished forecasts. To reconstruct a complete daily timeline for each contract, a Kalman filter was applied using a local linear trend state-space model (`SSModel`) from the `KFAS` package. The smoothed state estimates filled in the missing values, ensuring temporal continuity across contract boundaries.
  - *Financial Market Data:* Stock and ETF return data exhibited missing values corresponding to weekends and public holidays—non-trading days during which markets are closed. After verifying that these gaps aligned with expected calendar closures, no imputation was performed. Instead, `na.omit()` was applied to clean the dataset, ensuring that only valid trading sessions (typically five days per week) were retained for modeling purposes.

- **Data Integration:** Once the prediction signals and financial return series were processed, the two datasets were merged on the basis of the available trading dates from the financial market data (i.e., stock and ETF returns). This approach ensured temporal alignment and eliminated any potential look-ahead bias, by guaranteeing that only information available at or prior to each trading day was used for forecasting. By anchoring the integration to the financial data timeline, we avoided including future prediction signals that would not have been observable at the time of the actual market movement. The final dataset consisted of: (i) daily asset returns as target variables, (ii) corresponding prediction market signals, and (iii) metadata such as date, asset ticker, and event identifiers.

Date	**id**	**pred daily**	**ticker**	**returns**
2024-01-29	Tesla	-470.67267	AAPL	-1.458257e-03
2024-01-29	Tesla	-470.67267	AMZN	1.204971e-02
2024-01-29	Tesla	-470.67267	BAC	6.287425e-03
2024-01-29	Tesla	-470.67267	COIN	4.508616e-02
...	...	...	...	...
2024-01-30	Tesla	-838.51256	AAPL	-1.518802e-02
2024-01-30	Tesla	-838.51256	AMZN	-1.057872e-02
2024-01-30	Tesla	-838.51256	BAC	2.625369e-02
2024-01-30	Tesla	-838.51256	COIN	-3.844175e-02
...	...	...	...	...
2024-05-22	Netflix	0.029503250	AAPL	-0.0070995761
2024-05-22	Netflix	0.029503250	AMZN	-0.0040787470
2024-05-22	Netflix	0.029503250	BAC	0.0088809947
2024-05-22	Netflix	0.029503250	COIN	0.0288876050
...	...	...	...	...
2024-11-26	Apple	4.000	XOM	-0.0130511169
2024-11-26	Apple	4.000	SPY	0.0052214111
2024-11-26	Apple	4.000	IWDA.AS	0.0011463223
2024-11-26	Apple	4.000	QQQ	0.0053692356

To reinforce temporal causality and prevent future information from influencing model training or evaluation, the integration pipeline was anchored on financial market timestamps, ensuring that only contemporaneous or past Kalshi signals were matched with asset returns. In particular:

- Only prediction signals published prior to or on the same day as the asset return were used.
- Future data—even if technically available in the full dataset—was never leaked into the training or testing process.
- All model evaluations were conducted using walk-forward cross-validation, with strict temporal separation between training and test sets.

This careful design eliminated the risk of contaminating the model with information from the future and ensured the empirical validity of the forecasting exercise.

4.3 Global Modeling Preprocessing

In the global modeling framework (`global_model.R`), the objective was to uncover generalizable relationships between Kalshi forecast signals and financial asset returns by training a single unified model across all stocks. Unlike the individual asset modeling approach, this method used a common architecture to highlight predictive features consistent across companies, contracts, and time periods.

After completing the preprocessing and integration steps, the data were structured as a longitudinal panel dataset encompassing all tickers. Each row corresponded to a unique combination of trading date and stock ticker, with Kalshi signals serving as explanatory variables and daily returns as the target variable.

The forecast signals from Kalshi were transformed from long to wide format using the `pivot_wider()` function. Each distinct contract (e.g., `TSLA_Q1_prod`, `META_DAU`) was converted into a separate column, resulting in a high-dimensional feature space. Due to the specificity of some contracts to certain companies, missing values were retained and handled directly within the modeling process.

To enhance the model's capacity to learn company-specific dynamics, additional categorical metadata such as the stock ticker (`ticker`) was included as a predictor. These variables were converted into dummy variables using the `step_dummy()` function from the `recipes` package, allowing for the integration of company-level effects within a unified modeling structure.

The dataset was sorted chronologically and then divided into training and test sets, ensuring that the test data followed the training period to preserve temporal integrity. This procedure safeguarded against information leakage from the future into the training phase. All preprocessing operations—including imputation, encoding, and normalization—were conducted exclusively on the training set to uphold this separation.

Finally, missing values in the Kalshi signals, typically arising from the absence of relevant contracts for certain companies, were left as `NA`. The XGBoost algorithm was chosen in part for its ability to natively handle missing values, allowing the absence of information to be treated as a potentially informative feature.

This global modeling strategy leveraged cross-company variation while enforcing a robust temporal validation design. By training on a diverse, unified dataset that spanned various stocks and contract types, the model was positioned to identify generalizable and transferable patterns that improved both prediction accuracy and model applicability.

4.4 Local Modeling Preprocessing

The local modeling approach, implemented in the `local_model.R` script, focuses on creating individualized models for each asset, allowing for a more precise capture of asset-specific dynamics. This section outlines the key preprocessing steps applied within this localized framework.

For each asset (e.g., TSLA, NFLX, or META), the dataset is filtered to retain only the return series and the relevant Kalshi contracts specific to that stock. This filtering ensures that irrelevant signals are excluded, reducing noise and dimensionality for each model.

As local models are particularly sensitive to temporal properties, both the return series and prediction signals undergo stationarity tests. A custom function, `check_stationarity`, applies the Augmented Dickey-Fuller (ADF) test recursively, allowing for up to two differencing operations if necessary, to ensure the data satisfies stationarity requirements. If the initial series is non-stationary (p-value ¿ 0.05), the function applies differencing to achieve stationarity. This process ensures that methods dependent on stationarity are valid. The following R code illustrates the stationarity testing procedure:

```r
check_stationarity <- function(df, ticker) {
  df_clean <- df[!is.na(df$returns), ]
  adf_result <- adf.test(df_clean$returns, alternative = "stationary")

  if (adf_result\$p.value > 0.05) {
    df\$returns <- c(NA, diff(df\$returns, differences = 1))
    df_clean <- df[!is.na(df\$returns), ]
    adf_result_diff1 <- adf.test(df_clean\$returns, alternative = "stationary")

    if (adf_result_diff1\$p.value > 0.05) {
      df\$returns <- c(NA, diff(df\$returns, differences = 1))
    }
  }
  return(df)
}
```

In addition to stationarity checks, dynamic predictor selection is performed using Granger causality tests. These tests assess whether each differenced Kalshi signal provides statistically significant predictive information for the asset's return series. If a signal passes a predefined significance threshold (typically set at 5%), it is retained as a predictor. This step results in a refined and interpretable feature set for each asset. The following code snippet demonstrates how the Granger causality test is applied to identify significant predictors:

```r
granger_causality_test <- function(pred_series, stock_series, max_lag = 5) {
  df_combined <- inner_join(pred_series, stock_series, by = "date", suffix = c("_pred", "_stock"))
  df_clean <- df_combined %>% drop_na(pred_daily_pred, pred_daily_stock)
  grangertest(df_clean\$pred_daily_stock ~ df_clean\$pred_daily_pred, order = max_lag)
}
```

Unlike the global approach, the local models did not include any dummy variables for tickers or cross-sectional metadata, as each model pertained to a single stock by design. Instead, lagged features were engineered on a per-asset basis using functions from the `timetk` and `recipes` packages. This allowed each model to account for short-term memory and autoregressive effects specific to the asset in question.

Temporal integrity was maintained throughout the process by applying walk-forward cross-validation independently to each asset. Each fold respected chronological order, ensuring that validation sets always followed their corresponding training sets in time. Importantly, all preprocessing operations—including stationarity transformations and causality tests—were confined within each fold to prevent information leakage.

This localized modeling strategy enabled a more granular analysis of asset-specific behavior and improved prediction accuracy in contexts where global models might fail to capture nuanced patterns. Although computationally more demanding, this approach provided enhanced interpretability and robustness, especially in the face of heterogeneous predictive signal relevance across assets.

5 Modeling (Implementation)

This section details the practical implementation of the forecasting models. The primary global model implemented was XGBoost, and local models were ARIMAX, both evaluated within a walk-forward validation framework.

5.1 Technical specifications of the global model (XGBoost)

The global forecasting model used in this project is based on the XGBoost algorithm, a powerful gradient boosting technique particularly well-suited for structured/tabular data. The model is trained and evaluated using a time-series cross-validation framework with walk-forward validation to preserve the temporal structure of the data. Below are the key steps of the implementation:

Data Splitting and Cross-Validation: To evaluate the model's generalization ability, we use the `time_series_cv()` function to implement a walk-forward validation strategy. This method ensures that each training set contains only observations prior to the corresponding testing period. We use two non-overlapping folds with the following configuration:

```
# Creates two non-overlapping time splits with cumulative window
splits <- data_tbl %>%
  arrange(date) %>%
  time_series_cv(
    date_var   = date,
    initial    = 150 days,
    assess     = 75 days,
    skip       = 75 days,
    cumulative = TRUE,
    slice_limit = 2
  )
```

In this section of the code, we create a walk-forward validation strategy using the `time_series_cv()` function. First, we ensure that the data is ordered by date, which is essential for time-series analysis. The function splits the data into two folds with a training window of 150 days and a testing window of 75 days. The training window is cumulative, meaning each new fold includes all previous data. The `skip` argument is set to 75 days, meaning the training set for the next fold will start 75 days after the previous fold's test set. Finally, the `slice_limit` argument is used to limit the number of folds to 2 for simplicity.

Feature Engineering: The `recipe` function is used to preprocess the data for the XGBoost model. This includes several important steps. First, categorical variables (such as ”id”) are converted into dummy variables using `step_dummy()`. Numeric predictors are then normalized to have zero mean and unit variance using `step_normalize()`. Time-based features are extracted with `step_timeseries_signature()` to allow the model to learn seasonal patterns. The `step_rm()` function removes the ”date” column after the time-based features are extracted. Next, `step_zv()` removes predictors that have near-zero variance, which are unlikely to contribute to the model. Finally, `step_dummy()` is applied to nominal predictors to perform one-hot encoding, which transforms categorical variables into a format suitable for machine learning algorithms.

```
# Data preparation pipeline for XGBoost
rec_obj_xgb <- recipe(value ~ ., data = train_data) %>%
  step_dummy(id) %>%
  step_normalize(all_numeric(), -all_outcomes()) %>%
  step_timeseries_signature(date) %>%
  step_rm(date) %>%
  step_zv(all_predictors()) %>%
  step_dummy(all_nominal_predictors(), one_hot = TRUE)
```

Model Training: For each fold, an XGBoost regression model is trained within a `workflow` using the previously defined recipe. This workflow combines the data preprocessing steps with the model training.

```
# Creating a workflow combining the XGBoost model with the recipe
wflw_xgb <- workflow() %>%
  add_model(boost_tree("regression") %>%
  set_engine("xgboost")) %>%
  add_recipe(rec_obj_xgb) %>%
  fit(train_data)
```

In this part of the code, an XGBoost model is created using the `workflow()` function. This workflow integrates the pre-processing steps from the recipe with the XGBoost model. The model is specified as a regression model by using the `boost_tree()` function and setting the engine to `xgboost`. The `add_recipe()` function adds the preprocessing steps defined earlier, ensuring that the data is properly prepared before fitting the model. Finally, the model is trained using the `fit()` function on the training data.

Model Evaluation: The model is evaluated using the `modeltime` framework. For each fold, the calibrated model is assessed on unseen data and its accuracy is reported both globally and by ticker (i.e., for each time series). RMSE is used as the main metric to assess model performance.

```
# Calibrating the model on the test data
calib_tbl <- model_tbl %>%
  modeltime_calibrate(new_data = test_data, id = "ticker")

# Calculation of performance (RMSE, etc.) per ticker (id)
calib_tbl %>%
  modeltime_accuracy(acc_by_id = TRUE)
```

Once the model is trained, it is calibrated on the test data using the `modeltime_calibrate()` function. This function evaluates the model's performance on unseen data, and accuracy metrics (like RMSE) are calculated using the `modeltime_accuracy()` function. The evaluation is done both globally and by each ticker to assess the model's performance on individual time series.

Forecast Error Computation: Forecast errors $e_t = y_t - \hat{y}_t$ are computed and stored for further evaluation, combining both folds. This provides a comprehensive understanding of model performance over time, helping to identify patterns in prediction errors.

Final Forecasting and Refit: After validation, the model is refitted on the full dataset to produce future forecasts. A 30-day future frame is generated for each ticker, and forecasts with confidence intervals are produced.

```
# Retraining the model on the entire data set (after validation)
refit_tbl <- calib_tbl_1 %>%
  modeltime_refit(data = data_tbl)

forecast_results <- refit_tbl %>%
  modeltime_forecast(
    new_data = future_tbl,
    actual_data = data_tbl,
    conf_by_id = TRUE
  )
```

After model validation, the final model is refitted using the entire dataset with `modeltime_refit()`. This allows the model to incorporate all available data, including the validation data, before making future predictions. A 30-day future frame is generated for each ticker, and predictions are made using the `modeltime_forecast()` function, which also calculates confidence intervals for the forecasts.

Result Storage: Both RMSE metrics and final forecast outputs are saved as RDS files for reproducibility and further analysis. This ensures that results can be easily accessed and shared for future analysis or reporting.

```
# Saving RMSE results and forecasts
saveRDS(global_rmse_results, "data/rds/global_model_rmse.rds")

# Saving RMSE results for each ticker
saveRDS(forecast_results, "data/rds/global_model_forecasts.rds")
```

The results, including RMSE metrics and forecast outcomes, are saved in RDS files using the `saveRDS()` function. This ensures that the model's results can be reproduced in the future and shared for further analysis or reporting.

5.2 Technical specifications of the local model (ARIMAX)

To study the relationship between stock returns and prediction markets, we have implemented a local modeling approach using ARIMAX, which is an ARIMA model incorporating external explanatory variables. The approach is structured according to the following steps:

After verifying the stationarity of the data and the influence of the prediction markets on stock returns via the Granger causality test, we can now proceed to the implementation of the ARIMAX model.

5.2.1 Time Splitting: Walk-forward CV

We used a *walk-forward cross-validation* time splitting method to simulate a real forecasting situation. The dataset is divided into successive windows, with a training period of 150 days and a test period of 75 days. This procedure is repeated for two folds:

- **Fold 1:** Training data for the first 150 days, testing on the following 75 days.
- **Fold 2:** Cumulative training data (225 days), testing on the following 75 days.

```r
splits <- df_combined %>%
  arrange(date) %>%
  time_series_cv(
    date_var    = date,
    initial     = 150 days,
    assess      = 75 days,
    skip        = 75 days,
    cumulative  = TRUE,
    slice_limit = 2
  )
```

```r
first_split <- splits\$splits[[1]]
second_split <- splits\$splits[[2]]

train_data_1 <- analysis(first_split)
test_data_1  <- assessment(first_split)
train_data_2 <- analysis(second_split)
test_data_2  <- assessment(second_split)
```

5.2.2 Fitting the ARIMAX Model

Based on the results of the Granger test, two scenarios are considered:

- **No significant exogenous variables:** A simple ARIMA model is fitted.
- **At least one exogenous variable:** An ARIMAX model is fitted with the corresponding predictions as explanatory variables.

The exogenous variables are time-aligned with the return series and formatted into a matrix. Additional checks are performed to handle missing values and potential misalignments. In case of an error during fitting, the model defaults to a simple ARIMA model.

```r
fit_local_model_fold <- function(ticker_data, granger_results, prediction_markets, fold_number) {
  # Extraction of the current ticker
  current_ticker <- unique(ticker_data\$ticker)

  # Extraction of significant exogenous variables
  exog_vars <- granger_results %>%
    filter(Stock == current_ticker, result == "Granger-causes", p_value < 0.05) %>%
    pull(Prediction_Market)

  ts_data <- ts(ticker_data\$returns, frequency = 365)

  if (length(exog_vars) > 0) {
    exog_matrix <- prediction_markets %>%
      filter(id %in% exog_vars) %>%
      pivot_wider(names_from = id, values_from = pred_daily) %>%
      arrange(date) %>%
      filter(date %in% ticker_data$date) %>%
      select(-date) %>%
      as.matrix()

    if (any(is.na(exog_matrix))) {
      model <- auto.arima(ts_data)
    } else {
      model <- auto.arima(ts_data, xreg = exog_matrix)
    }
  } else {
    model <- auto.arima(ts_data)
  }

  return(model)
}
```

6 Analysis of Results

6.1 Implementation of ARIMA and ARIMAX Models

6.1.1 Stationarity Assessment

To ensure that the time series models met the assumptions of stationarity, the Augmented Dickey-Fuller (ADF) test was applied to all time series involved in the forecasting models.

First, the financial time series — specifically, the daily stock return series — were tested. The results confirmed that all return series were already stationary at level. Therefore, no differencing was required before modeling, which is consistent with the literature where stock returns typically exhibit mean-reverting behavior and no unit root.

Second, the same test was applied to the prediction market signals. The ADF test results indicated that the majority of these signals were non-stationary in their raw form. Consequently, first-order differencing was applied to achieve stationarity. Specifically, 11 out of the 14 prediction signals required one differencing step. These included forecasts related to Tesla, Netflix, U.S. GDP, SpaceX launches, U.S. gas prices, Bitcoin (BTC), U.S. semiconductor demand, inflation expectations, hurricane counts, Ethereum (ETH), and measles incidence.

By contrast, three prediction market signals — Meta daily active users (META), WTI oil prices, and Apple car development — were found to be stationary at level, requiring no transformation.

This preprocessing step was essential to ensure the validity of the ARIMA and ARIMAX models, as non-stationary inputs can bias the estimates and lead to misleading inferences. Table 1 summarizes the number of differentiations applied to each prediction market variable.

Pred Market	Number of differentiations
TESLA	1
NETFLIX	1
GDP	1
SpaceX	1
Gas US	1
BTC	1
US Semi Conductor	1
Inflation	1
Hurricanes	1
ETH	1
Measles	1
META	0
WTI Oil	0
Apple	0

Figure 1:

6.1.2 Granger Causality Results

To identify which prediction market signals held predictive power over specific stock or ETF returns, pairwise Granger causality tests were conducted at a 5% significance level. These tests were applied to the differenced time series when required, as discussed in the previous subsection.

The results indicate that a subset of prediction market variables exhibited statistically significant Granger-causal relationships with financial assets. Figure 2 summarizes all tested relationships, distinguishing between those that are statistically significant (p ¡ 0.05) and those that fall within the marginal range (0.05 ¡ p ¡ 0.10).

These findings suggest that signals related to macroeconomic activity (e.g., GDP forecasts), health events (e.g., measles incidence), and company-specific expectations (e.g., Netflix or Meta) can provide predictive insights into stock or ETF returns. Notably, the GDP prediction market signal was the most frequently significant, Granger-causing six different assets.

Although variables in the 0.05–0.10 range were not retained for ARIMAX modeling to maintain statistical rigor, they may nonetheless carry weak predictive signals that warrant further investigation in future studies.

Pred Market	Stock	P-value	Result
Meta	JNJ	0.0061	Granger-causes
Hurricanes	GOOG	0.0081	Granger-causes
Measles	NFLX	0.0203	Granger-causes
GDP	IWDA.AS	0.0233	Granger-causes
SpaceX	NFLX	0.0281	Granger-causes
GDP	XOM	0.0307	Granger-causes
GDP	SPY	0.0386	Granger-causes
GDP	WMT	0.0471	Granger-causes
Hurricanes	TSLA	0.0485	Granger-causes
Meta	PFE	0.0530	Does not Granger-cause
Apple	WMT	0.0551	Does not Granger-cause
ETH	MSFT	0.0574	Does not Granger-cause
GDP	QQQ	0.0696	Does not Granger-cause
SpaceX	WMT	0.0796	Does not Granger-cause
BTC	SPY	0.0872	Does not Granger-cause
Netflix	PFE	0.0927	Does not Granger-cause

Figure 2:

6.2 Comparative Performance (RMSE)

6.2.1 Performance on Fold 1

To evaluate the predictive performance of the global versus local modeling strategies, the Root Mean Squared Error (RMSE) was computed for each asset over the first walk-forward validation fold. Figure 3 presents the RMSE results for both models, alongside an indication of which model performed better for each asset.

The local models outperformed the global model for the vast majority of assets. Specifically, the ARIMA or ARIMAX models achieved lower RMSE values in 16 out of 19 cases. This supports the hypothesis that asset-specific modeling may better capture individual return dynamics, especially when tailored with causally relevant exogenous variables.

However, there were a few exceptions. For COIN, NFLX, and XOM, the global model achieved equal or slightly better RMSE. These cases may reflect stocks for which common patterns captured by the global model (e.g., via tree-based interactions) were more predictive than individual historical or causal relationships.

Overall, these results suggest that while global models offer efficiency and generalizability, local models may yield superior forecasting accuracy when sufficient data and relevant exogenous signals are available.

Ticker	Global RMSE	Local RMSE	Better Model
AAPL	0.0090	0.0083	Local
AMZN	0.0154	0.0124	Local
BAC	0.0114	0.0112	Local
COIN	0.0477	0.0485	Global
GOOG	0.0138	0.0138	Local
INTC	0.0220	0.0212	Local
IWDA.AS	0.0083	0.0073	Local
JNJ	0.0112	0.0093	Local
JPM	0.0137	0.0129	Local
META	0.0143	0.0126	Local
MSFT	0.0117	0.0098	Local
NFLX	0.0141	0.0148	Global
NVDA	0.0300	0.0196	Local
PFE	0.0131	0.0130	Local
QQQ	0.0113	0.0096	Local
SPY	0.0079	0.0070	Local
TSLA	0.0275	0.0244	Local
WMT	0.0107	0.0092	Local
XOM	0.0116	0.0116	Global

Figure 3:

6.2.2 Performance on Fold 2

The second fold further reinforces the results observed in the first iteration. As shown in Figure 4, local models outperformed global models in 18 out of 19 cases. This consistent advantage highlights the robustness of asset-specific modeling strategies.

The only exception in this fold was `MSFT`, where the global model outperformed the local model. However, this may be attributable to a temporary overfitting of the local model or instability in the model selection process due to limited data in that fold. Aside from this outlier, the superiority of the local approach remains strong and consistent.

Ticker	Global RMSE	Local RMSE	Better Model
AAPL	0.0140	0.0128	Local
AMZN	0.0198	0.0157	Local
BAC	0.0145	0.0129	Local
COIN	0.0511	0.0477	Local
GOOG	0.0157	0.0147	Local
INTC	0.0283	0.0275	Local
IWDA.AS	0.0109	0.0097	Local
JNJ	0.0106	0.0102	Local
JPM	0.0151	0.0140	Local
META	0.0211	0.0197	Local
MSFT	0.0130	0.0225	Global
NFLX	0.0152	0.0145	Local
NVDA	0.0367	0.0358	Local
PFE	0.0128	0.0117	Local
QQQ	0.0164	0.0158	Local
SPY	0.0116	0.0107	Local
TSLA	0.0382	0.0357	Local
WMT	0.0091	0.0085	Local
XOM	0.0133	0.0123	Local

Figure 4:

6.3 Statistical Significance (Diebold-Mariano Test)

To assess whether the observed differences in forecast accuracy between the global and local models were statistically significant, the Diebold-Mariano (DM) test was performed for each asset and each fold. This test compares the forecast errors of two predictive models over the same time window and evaluates whether their differences are distinguishable from zero.

Table 5 summarizes the results, including the DM test statistic, p-value, and the conclusion regarding model superiority for each ticker and fold. A result is considered statistically significant if the p-value is below 0.05.

Across the 38 test cases (19 assets $\times$ 2 folds), the local models were statistically better in **6 cases**, while the global model was significantly better in **1 case** (`MSFT`, Fold 2). In all remaining cases, the DM test did not detect a statistically significant difference in forecasting accuracy at the 5% level.

Ticker	Fold	DM Statistic	P-value	Conclusion	N Observations
AAPL	Fold 1	0.9791	0.3321	No difference	53
AAPL	Fold 2	1.2352	0.2225	No difference	51
AMZN	Fold 1	2.1936	0.0328	Local ✓	53
AMZN	Fold 2	3.1257	0.0030	Local ✓	51
BAC	Fold 1	0.4777	0.6349	No difference	53
BAC	Fold 2	2.0583	0.0448	Local ✓	51
COIN	Fold 1	-0.4358	0.6648	No difference	53
COIN	Fold 2	1.4593	0.1507	No difference	51
GOOG	Fold 1	0.0813	0.9355	No difference	51
GOOG	Fold 2	2.4301	0.0187	Local ✓	51
INTC	Fold 1	0.5021	0.6178	No difference	51
INTC	Fold 2	0.8389	0.4055	No difference	51
IWDA.AS	Fold 1	1.1126	0.2710	No difference	53
IWDA.AS	Fold 2	1.7427	0.0873	No difference	53
JNJ	Fold 1	4.0505	0.0002	Local ✓	51
JNJ	Fold 2	0.5103	0.6121	No difference	51
JPM	Fold 1	1.1543	0.2539	No difference	51
JPM	Fold 2	1.7547	0.0854	No difference	51
META	Fold 1	2.0198	0.0488	Local ✓	51
META	Fold 2	1.2666	0.2112	No difference	51
MSFT	Fold 1	2.7692	0.0079	Local ✓	51
MSFT	Fold 2	-8.1694	0.0000	Global ✓	51
NFLX	Fold 1	-0.9318	0.3559	No difference	51
NFLX	Fold 2	1.1179	0.2689	No difference	51
NVDA	Fold 1	3.8234	0.0004	Local ✓	51
NVDA	Fold 2	0.8141	0.4194	No difference	51
PFE	Fold 1	-0.1201	0.9049	No difference	51
PFE	Fold 2	1.3527	0.1822	No difference	51
QQQ	Fold 1	1.8467	0.0705	No difference	53
QQQ	Fold 2	1.2531	0.2160	No difference	51
SPY	Fold 1	1.4017	0.1669	No difference	53
SPY	Fold 2	1.3091	0.1965	No difference	51
TSLA	Fold 1	1.8633	0.0683	No difference	51
TSLA	Fold 2	1.4134	0.1637	No difference	51
WMT	Fold 1	1.6203	0.1112	No difference	53
WMT	Fold 2	0.7302	0.4687	No difference	51
XOM	Fold 1	-0.2090	0.8353	No difference	53
XOM	Fold 2	1.4613	0.1502	No difference	51

Figure 5:

6.4 Hypothesis Evaluation

This section revisits the research hypotheses in light of the empirical findings:

- **H1:** *Prediction market signals provide additional information for stock return forecasting.*
  Supported for several stocks where ARIMAX outperformed ARIMA and had lower RMSE values, suggesting that Kalshi signals are informative.

- **H2:** *Prediction market signals are more effective when used in local models.*
  Partially supported. While local models using exogenous variables often performed better, the global model (XGBoost) remained competitive on some stocks.

- **H3:** *The influence of prediction market signals varies by stock.*
  Clearly supported. Granger causality tests show heterogeneity in predictive relationships across stocks.

7 Discussion

This section compares the results obtained with those of previous studies and outlines the practical and theoretical implications of the study. It also explores potential limitations and alternative interpretations of the findings.

7.1 Methodological Insights

- Forces et faiblesses des approches globales vs locales
- Leçons apprises sur la modélisation des séries temporelles financières
- Implications pour la recherche future en finance quantitative

7.2 Limitations and Challenges

- Limitations des données
- Contraintes méthodologiques
- Défis pratiques dans l'implémentation

8 Conclusion

This final section summarizes the main findings of the study, discusses its limitations and suggests avenues for future research.

Bibliography

Ah Mand, A., Janor, H., Abdul Rahim, R., & Sarmidi, T. (2023). Herding behavior and stock market conditions. *PSU Research Review*, *7*(2), 105–116. https://doi.org/10.1108/PRR-10-2020-0033

Berg, J. E., Nelson, F. D., & Rietz, T. A. (2008). Prediction market accuracy in the long run. *International Journal of Forecasting*, *24*(2), 285–300. https://doi.org/10.1016/j.ijforecast.2008.03.007

Berg, J. E., Neumann, G. R., & Rietz, T. A. (2009). Searching for Google's Value: Using Prediction Markets to Forecast Market Capitalization Prior to an Initial Public Offering [Publisher: INFORMS]. *Management Science*, *55*(3), 348–361. Retrieved January 14, 2025, from https://www.jstor.org.proxy.bib.uclouvain.be:8888/stable/40539152

Bossaerts, F., Yadav, N., Bossaerts, P., Nash, C., Todd, T., Rudolf, T., Hutchins, R., Ponsonby, A.-L., & Mattingly, K. (2022, September). Price Formation in Field Prediction Markets: The Wisdom in the Crowd [arXiv:2209.08778 [q-fin]]. https://doi.org/10.48550/arXiv.2209.08778

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: Forecasting and control* (5th). John Wiley & Sons.

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*, 785–794. https://doi.org/10.1145/2939672.2939785

Dai, M., Jia, Y., & Kou, S. (2021). The wisdom of the crowd and prediction markets. *Journal of Econometrics*, *222*(1), 561–578. https://doi.org/10.1016/j.jeconom.2020.07.016

Downey, l. (2024, October). Efficient Market Hypothesis (EMH): Definition and Critique. Retrieved January 15, 2025, from https://www.investopedia.com/terms/e/efficientmarkethypothesis.asp

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural computation*, *4*(1), 1–58.

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

Hartford, J., Leyton-Brown, K., Ravanbakhsh, S., & Poole, D. (2018). Deep learning for predicting asset returns. *arXiv preprint arXiv:1804.09314*.

Keasey, K., Lambrinoudakis, C., Mascia, D. V., & Zhang, Z. (2024). The impact of social media influencers on the financial market performance of firms. *European Financial Management*, eufm.12513. https://doi.org/10.1111/eufm.12513

Li, Q., Wang, T., Li, P., Liu, L., Gong, Q., & Chen, Y. (2014). The effect of news and public mood on stock movements. *Information Sciences*, *278*, 826–840. https://doi.org/10.1016/j.ins.2014.03.096

Lopez de Prado, M. (2018). *Advances in financial machine learning*. John Wiley & Sons.

Mongrain, P., & Stegmaier, M. (2024). Introduction to Forecasting the 2024 US Elections. *PS: Political Science & Politics*, 1–12. https://doi.org/10.1017/S1049096524001008

Montero-Manso, P., & Hyndman, R. J. (2021). Modelling global versus local time series using hierarchical forecasting models. *International Journal of Forecasting*, *37*(1), 117–134.

Naseer, M. (2016). The Efficient Market Hypothesis: A Critical Review of the Literature. *The IUP Journal of Financial Risk Management*, *Vol. XII*.

Nti, I. K., Adekoya, A. F., & Weyori, B. A. (2020). A systematic review of fundamental and technical analysis of stock market predictions. *Artificial Intelligence Review*, *53*(4), 3007–3057. https://doi.org/10.1007/s10462-019-09754-z

Tashman, L. J. (2000). Out-of-sample tests of forecasting accuracy: An analysis and review. *International journal of forecasting*, *16*(4), 437–450.

Waitz, M., & Mild, A. (2013). Corporate prediction markets: A tool for predicting market shares. *Journal of Business Economics*, *83*(3), 193–212. https://doi.org/10.1007/s11573-012-0645-1

Wolfers, J., & Zitzewitz, E. (2004). Prediction Markets. *Journal of Economic Perspectives*, *18*(2), 107–126. https://doi.org/10.1257/0895330041371321

Appendix

Abstract :

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
Louvain School of Management
Place des Doyens, 1 bte L2.01.01, 1348 Louvain-la-Neuve
Boulevard Emile Devreux 6, 6000 Charleroi, Belgique
Chaussée de Binche 151, 7000 Mons, Belgique
www.uclouvain.be/lsm