Machine Learning

Capucine Wyseur, Emile Sagot, Chiara Godet

December 2025

Table of Contents

1 Introduction and Business Case

Financial markets are often described as efficient, meaning that asset prices fully reflect all available information. Under this assumption, short-term stock price movements should follow a random walk and should not be predictable. However, many empirical studies suggest that financial markets can show anomalies, non-linear patterns, and short-term dependencies.

The objective of this project is to study whether short-term stock market returns can be predicted using machine learning methods. More precisely, the goal is to test if non-linear technical indicators, combined with supervised learning algorithms, can help anticipate movements of the S&P 500 index.

The central research question of this study is the following:

> Can short-term stock market returns be effectively modeled and predicted by combining non-linear technical indicators with supervised machine learning algorithms?

This problem is directly related to financial engineering and quantitative finance, where forecasting, risk management, and trading strategies rely on extracting useful information from market data. Even small predictive signals can be valuable in short-term trading contexts.

A major challenge of this problem is the risk of overfitting. Financial data are noisy, non-stationary, and highly complex. Models that perform well on historical data may fail when applied to new data. For this reason, rigorous out-of-sample validation is required to ensure that results are robust and generalizable.

This project therefore focuses not only on model performance, but also on proper validation methods, realistic assumptions, and careful evaluation of predictive power under real market conditions.

2 Dataset Description and Data Source

The datasets used in this project were obtained from Yahoo Finance, a public and widely used platform that provides historical financial market data. Data were collected using the `yfinance` Python library, which allows direct and reproducible access to Yahoo Finance databases.

Three major stock market indices were downloaded: the S&P 500, the Dow Jones Industrial Average (DJ30), and the CAC 40. Each index was saved in a separate CSV file. Although several indices were collected, the S&P 500 dataset is the main focus of this study and is used for the core analysis and modeling.

The data covers the period from January 2010 to January 2025, with a daily frequency. Each row of the dataset corresponds to one trading day, which makes the data suitable for short-term market analysis.

For each index, the following variables were extracted directly from Yahoo Finance:

- **Open**: opening price of the index on the trading day
- **High**: highest price reached during the day
- **Low**: lowest price reached during the day
- **Close**: closing price of the index
- **Volume**: total trading volume

All variables are numerical and indexed by date. The datasets are structured as time series, preserving the chronological order of observations.

Basic preprocessing steps were applied to ensure data consistency. Date fields were converted into proper datetime formats, column types were checked and converted when necessary, and the data was cleaned to remove non-relevant or improperly formatted columns. The final datasets contain continuous daily observations without structural breaks.

All data used in this project comes from publicly available sources and does not include any confidential or proprietary information.

3 Data Preprocessing and Feature Construction

This section describes the data preprocessing steps and the construction of technical indicators used in this project. These steps are essential to transform raw market data into a clean and structured dataset suitable for supervised machine learning.

3.1 Data Cleaning and Formatting

The S&P 500 dataset was first loaded from a CSV file and carefully cleaned to ensure consistency. Date information was properly converted into a datetime format and set as the index of the dataset in order to preserve the chronological structure of the time series.

All price and volume variables were converted into numerical types. This step ensures that mathematical operations and indicator calculations can be applied correctly. The dataset was also checked for missing or invalid values. Any rows affected by shifts or rolling computations were removed to maintain data integrity.

After preprocessing, the dataset contains daily observations from January 2010 to December 2024, with five raw market variables: Open, High, Low, Close, and Volume.

3.2 Returns and Target Variable Construction

To study short-term price movements, daily returns were computed using percentage changes in closing prices. These returns capture relative price variations and are commonly used in financial modeling.

The prediction task is formulated as a binary classification problem. The target variable represents the direction of the next day's return:

- A value of 1 indicates a positive return
- A value of 0 indicates a negative or zero return

The target variable is shifted forward in time so that only past information is used to predict future outcomes. This approach strictly avoids any form of look-ahead bias.

3.3 Technical Indicators Overview

To capture non-linear market behavior, several technical indicators were constructed from historical price and volume data. These indicators aim to summarize momentum, trend, and price acceleration in a compact form. All indicators were shifted by one day to ensure that only information available at time t is used to predict returns at time $t + 1$.

3.4 Moving Average Convergence Divergence (MACD)

The Moving Average Convergence Divergence (MACD) is a momentum-based indicator designed to capture changes in trend direction and strength. It is computed as the difference between a short-term and a long-term exponential moving average of prices. A signal line and a histogram are derived to highlight momentum shifts.

In this project, the MACD line, the signal line, and the histogram are used as separate explanatory variables.

The MACD plot shows alternating positive and negative phases over time, reflecting changes in market momentum. Periods where the MACD line diverges from the signal line correspond to trend accelerations, while histogram contractions indicate momentum slowdowns. This confirms that the indicator reacts dynamically to changes in price movements.

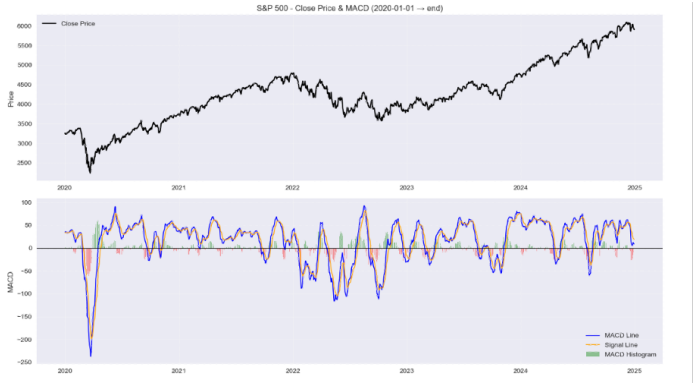


Figure 1: MACD plot for the S&P 500

3.5 Relative Strength Index (RSI)

The Relative Strength Index (RSI) is a bounded momentum oscillator that ranges between 0 and 100. It measures the speed and magnitude of recent price movements and is commonly used to identify overbought and oversold market conditions.

In this project, a 14-day RSI is computed. High RSI values indicate strong upward momentum, while low values suggest downward pressure or potential exhaustion of a trend.

The RSI indicator oscillates between 0 and 100, with several episodes approaching overbought and oversold zones. These movements illustrate how the RSI captures short-term momentum fluctuations and potential exhaustion phases.

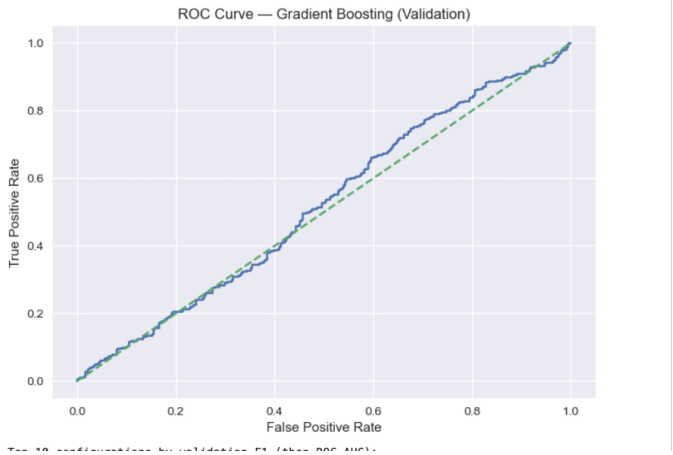


Figure 2: Enter Caption

3.6 Moving Averages and Momentum Indicators

Both Simple Moving Averages (SMA) and Exponential Moving Averages (EMA) were computed to smooth price fluctuations and identify underlying trends. Differences between prices and moving averages were also calculated to quantify deviations from trend levels.

Additionally, momentum indicators were computed over multiple horizons (10-day and 20-day). These indicators measure the rate of price change and help capture acceleration or deceleration in market movements.

Moving averages smooth short-term price fluctuations and highlight medium-term trends. Momentum indicators vary around zero and reflect changes in the speed of price movements over different horizons.

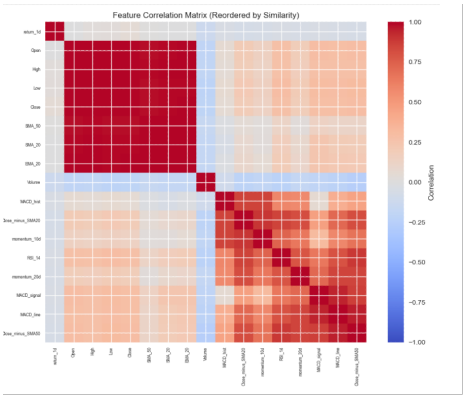
3.7 Time-Aware Data Splitting

To ensure a realistic evaluation of model performance, the dataset was split using a strictly time-based structure. No random shuffling was applied in order to preserve the chronological nature of financial data and to avoid any form of information leakage.

The data was divided into three distinct periods:

- Training period: 2015–2019
- Validation period: 2020–2022
- Test period: 2023–2024

This temporal separation mimics real-world trading conditions, where models are trained on historical data and evaluated on unseen future observations. It ensures that model performance reflects a genuine out-of-sample setting.



The correlation matrix reveals clusters of highly related features, particularly among trend-based indicators. This visualization helps identify potential redundancy and supports later feature selection decisions.

3.8 Feature Correlation Analysis

A correlation analysis was conducted on the training dataset to examine relationships between the constructed technical indicators. This analysis helps identify dependencies between features and provides insight into how different indicators interact.

The correlation structure reveals clusters of highly related variables, particularly among trend-based indicators such as moving averages and price-to-average

deviations. Momentum-related indicators also exhibit moderate correlations across similar horizons.

Hierarchical clustering was applied to reorder the correlation matrix based on feature similarity. This reordering improves interpretability and helps highlight potential redundancy among indicators.

At the end of the preprocessing stage, the final dataset consists of:

- Clean and consistent time-series data
- A binary target variable suitable for classification
- Multiple technical indicators capturing trend, momentum, and non-linear dynamics
- Strict avoidance of look-ahead bias

This processed dataset serves as the input for the supervised machine learning models presented in the next section.

4 Problem Definition

The objective of this project is to study whether short-term stock market movements can be anticipated using machine learning techniques. Financial markets are often considered difficult to predict at very short horizons, especially on a day-to-day basis. Price variations are noisy and strongly influenced by new information that is not always observable in advance.

In this project, the problem is formulated as a binary classification task. Instead of predicting the exact value of future returns, the goal is to predict their direction. More precisely, the task consists in determining whether the next-day return of the S&P 500 index will be positive or not, using only past market information.

The explanatory variables are based on historical prices and technical indicators such as moving averages, momentum indicators, MACD, and RSI. These indicators are commonly used in technical analysis and are designed to summarize past market behavior in a compact and interpretable way.

The prediction is performed in a strictly time-aware setting. Only information available up to time t is used to predict the market movement at time $t + 1$. This approach avoids any look-ahead bias and reflects realistic trading conditions.

The central question addressed by this project is therefore the following:

Can short-term market direction be predicted using technical indicators and supervised machine learning models, and can this information be exploited in a simple trading strategy?

This formulation aligns the modeling task, the evaluation metrics, and the trading strategy with realistic financial constraints. Rather than aiming for perfect prediction accuracy, the objective is to assess whether machine learning can extract weak but useful signals from historical market data.

5 Model Presentation

This section presents the supervised learning models used to predict the next-day direction of S&P 500 returns. The goal is to evaluate whether technical indicators and price-based features contain useful information for short-term market direction prediction.

The problem is formulated as a binary classification task. The target variable is defined as the sign of the next-day return. A value of 1 corresponds to a positive return, while a value of 0 corresponds to a negative or zero return. This formulation allows the use of standard classification algorithms and performance metrics.

To ensure a realistic evaluation, all models are trained and tested using a time-aware split. The dataset is divided into training, validation, and test periods without any random shuffling. This approach respects the chronological nature of financial data and avoids look-ahead bias. It mimics real trading conditions where future data is not available at training time.

All models rely on the same set of explanatory variables. These include raw market variables (Open, High, Low, Close, and Volume), lagged returns, and the technical indicators constructed during the preprocessing stage. The target variables are explicitly excluded from the feature matrix to prevent leakage.

5.1 Baseline Model: Logistic Regression

Logistic Regression is first used as a baseline model. Its main advantage is its simplicity and interpretability. As a linear classifier, it provides a clear reference point against which more complex non-linear models can be compared.

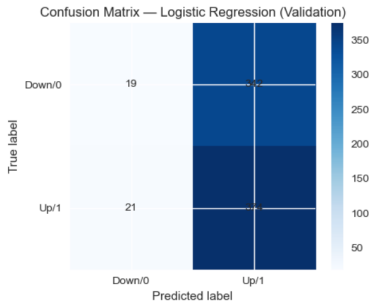
Because Logistic Regression is sensitive to differences in feature scale, a Pipeline is used to combine data standardization and model estimation. Feature scaling is fitted only on the training data and then applied to the validation and test sets. This ensures that no information from future data leaks into the training process.

On the validation set, Logistic Regression achieves moderate performance. The accuracy is slightly above 50%, which is only marginally better than random guessing. However, the model shows a very high recall for positive returns, meaning that it predicts upward movements in most cases. At the same time, it struggles to correctly identify negative returns, which results in poor recall for class 0.

This behavior is clearly visible in the confusion matrix, where most observations are classified as positive. This highlights an important limitation of simple linear models in this context: they tend to be biased toward the majority or easier-to-predict class. As a result, accuracy alone is not sufficient to assess model quality, and metrics such as precision, recall, and F1-score become essential.

One advantage of Logistic Regression is that it allows direct interpretation of feature coefficients. The largest coefficients are associated with variables such as price levels, moving-average deviations, and MACD components. This suggests

that trend and momentum-related features play a meaningful role in the model's decisions, even in a linear setting.



The confusion matrix for Logistic Regression shows a very asymmetric prediction behavior. The model correctly predicts most positive return days, which explains the very high recall for class 1. However, it fails to correctly identify negative return days, as shown by the large number of false positives.

This confirms that the Logistic Regression model is strongly biased toward predicting upward market movements. While this behavior slightly improves overall accuracy compared to random guessing, it leads to poor discrimination between positive and negative returns.

From a financial perspective, this result highlights the difficulty of capturing downside movements using a simple linear model. It also shows that accuracy alone is not sufficient to evaluate performance in financial classification tasks, since a biased model can appear acceptable while failing to detect important risk events.

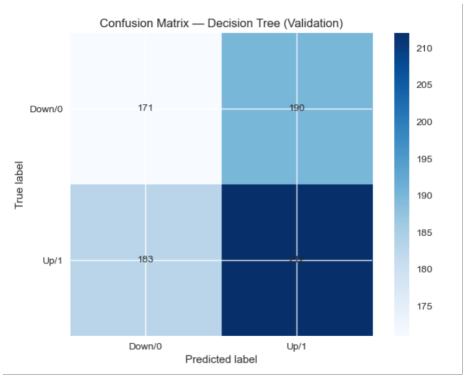
5.2 Non-linear Baseline: Decision Tree

To introduce non-linearity, a Decision Tree classifier is used as a second baseline model. Decision Trees can capture threshold effects and interactions between features without requiring any scaling. This makes them suitable for modeling non-linear patterns in financial data.

The default Decision Tree shows slightly weaker performance than Logistic Regression in terms of F1-score. While it achieves a more balanced precision and recall across classes, its overall predictive power remains limited.

An important advantage of Decision Trees is that they clearly illustrate the problem of overfitting. By increasing the maximum depth of the tree, training performance improves rapidly, while validation performance does not. This gap between training and validation results becomes more pronounced for deeper trees.

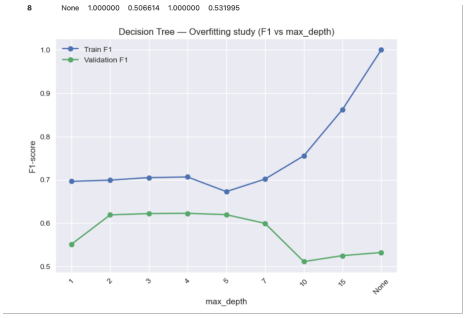
An overfitting study based on the maximum depth parameter confirms this behavior. The best validation F1-score is obtained with a relatively shallow tree. This shows that controlling model complexity is necessary to preserve generalization and avoid fitting noise in historical data.



The confusion matrix of the Decision Tree model shows a more balanced distribution between predicted classes compared to Logistic Regression. Both positive and negative returns are predicted with similar frequency, which explains the more balanced precision and recall values.

However, overall performance remains limited. The model does not significantly improve classification accuracy or F1-score. This indicates that although Decision Trees can model non-linear relationships, a single tree lacks robustness when applied to noisy financial data.

This result illustrates an important property of financial time series: capturing non-linear patterns is necessary but not sufficient. Without proper regularization, models can easily fit noise instead of true predictive signals.



The overfitting plot clearly shows that as the maximum depth of the Decision Tree increases, training performance improves rapidly while validation performance stagnates or declines.

This divergence between training and validation F1-scores is a textbook example of overfitting. Deeper trees memorize historical patterns but fail to generalize to unseen data.

The optimal depth corresponds to a relatively shallow tree, confirming that strong regularization is required when modeling financial markets. This result reinforces the project's central challenge: extracting weak predictive signals without fitting noise.

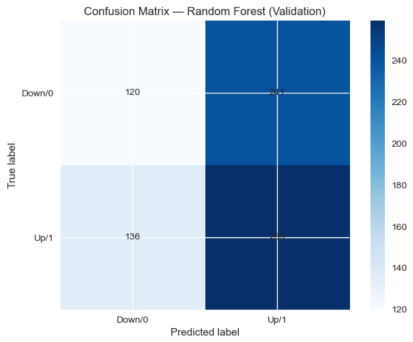
5.3 Ensemble Model: Random Forest

Random Forest is introduced as a more robust non-linear model. It combines multiple Decision Trees trained on different subsets of data and features. By averaging the predictions of many trees, Random Forest reduces variance and improves stability compared to a single tree.

The baseline Random Forest model already shows improved recall and F1-score compared to the previous models. A small hyperparameter search is then conducted to further improve performance. The best configuration uses a limited tree depth and a minimum number of observations per leaf, which helps control overfitting.

With these settings, Random Forest achieves the highest validation F1-score among all tested models. This indicates that ensemble methods are better suited to capture weak and noisy predictive signals in financial time series.

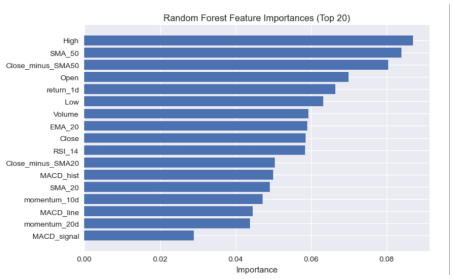
Another important advantage of Random Forest is the ability to compute feature importances. The most influential variables include price-based features, moving-average deviations, and return-related variables. This confirms that both raw price information and engineered technical indicators contribute to the model's predictions.



The Random Forest confusion matrix shows a better balance between false positives and false negatives compared to both Logistic Regression and Decision Trees. This leads to a higher F1-score and improved stability across classes.

This improvement confirms the advantage of ensemble methods in financial prediction tasks. By aggregating many weak learners, Random Forest reduces variance and captures more robust patterns in the data.

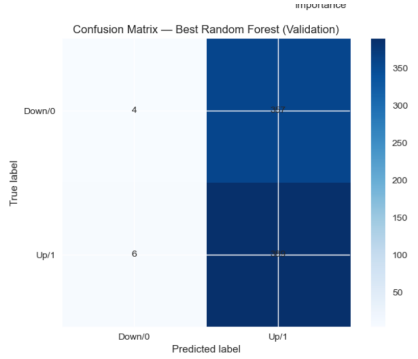
However, misclassifications remain frequent, indicating that even ensemble models struggle to fully separate positive and negative next-day returns. This highlights the inherently low signal-to-noise ratio in short-term market prediction.



The feature importance plot reveals that price-based variables, moving-average deviations, and lagged returns are among the most influential features. Momentum indicators and MACD-related variables also contribute meaningfully.

This result aligns well with financial intuition and the project's hypothesis. It confirms that technical indicators capturing trend persistence and short-term momentum are more informative than raw volume alone.

At the same time, the absence of a single dominant feature suggests that predictive power is distributed across many weak signals rather than driven by one strong indicator. This explains why complex models are required but still achieve only moderate performance.



The confusion matrix for Gradient Boosting shows slightly improved balance compared to previous models, with better detection of positive returns and moderate control of false positives.

Nevertheless, classification errors remain substantial. This indicates that even sequential ensemble methods, which are more flexible than Random Forests, cannot fully overcome the randomness present in short-term market movements.

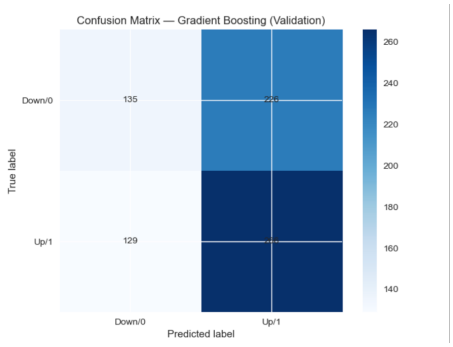
This result suggests that model complexity alone cannot solve the prediction problem and that the limits are largely imposed by market efficiency.

5.4 Boosted Trees: Gradient Boosting

Gradient Boosting is the most advanced model considered in this study. Unlike Random Forest, which trains trees independently, Gradient Boosting builds trees sequentially, with each new tree correcting the errors of the previous ones. This allows the model to capture more complex patterns.

The baseline Gradient Boosting model slightly improves accuracy compared to other methods, but overall performance remains modest. A hyperparameter search is performed to tune the number of trees, learning rate, and tree depth. The best configuration leads to improved F1-score but only a small increase in ROC-AUC.

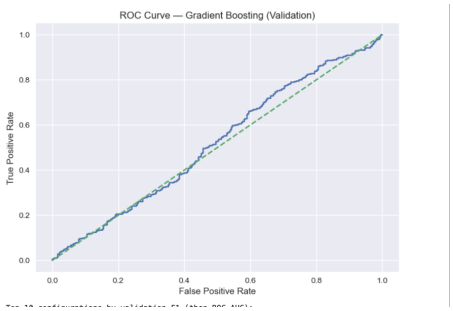
The ROC curve provides additional insight into the model's probabilistic performance. The ROC-AUC values remain close to 0.5, which suggests that even the most advanced models struggle to clearly separate positive and negative returns. This result highlights the intrinsic difficulty of short-term market direction prediction and the weak signal-to-noise ratio in financial data.



The confusion matrix shows that the Gradient Boosting model predicts the Up (1) class more frequently than the Down (0) class. A large number of positive return days are correctly classified, which explains the relatively high recall for class 1.

However, many negative return days are incorrectly predicted as positive. This leads to a significant number of false positives and weak performance on class 0. As a result, the model struggles to correctly identify downward market movements.

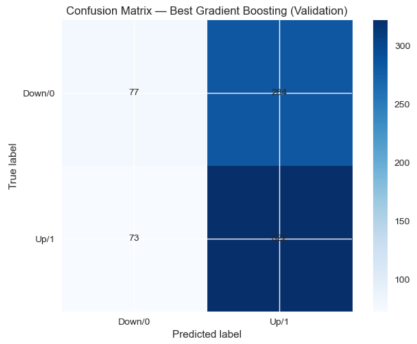
From a financial perspective, this behavior is problematic because negative returns often correspond to risk events. While the model captures some upward momentum, it fails to provide reliable downside protection. This confirms that short-term market direction remains difficult to predict, even with advanced ensemble methods.



The ROC curve lies very close to the diagonal line, and the corresponding ROC-AUC value is close to 0.5. This indicates that the model's ability to distinguish between positive and negative returns is only slightly better than random guessing.

This result shows that, although Gradient Boosting improves classification metrics such as recall and F1-score, its probabilistic ranking power remains weak. The model does not consistently assign higher probabilities to true positive returns than to negative ones.

In the context of this project, this confirms that short-term stock market movements contain very limited exploitable information. Even when non-linear interactions are modeled, the underlying signal-to-noise ratio remains low.



After hyperparameter tuning, the Gradient Boosting model shows a modest improvement in F1-score compared to the baseline configuration. This indicates that tuning the number of estimators, learning rate, and tree depth helps the model better fit the structure of the data.

However, the improvement in ROC-AUC remains very limited. This suggests that the gains mainly come from threshold effects in classification rather than from a true improvement in probabilistic discrimination.

This result highlights an important limitation of advanced machine learning models in financial applications: increasing model complexity can improve some metrics, but it does not necessarily lead to better generalization or stronger predictive signals. The market remains largely unpredictable at very short horizons.

5.5 Summary of Model Performance

In summary, four classification models were evaluated under a strict time-aware validation framework. Logistic Regression serves as an interpretable baseline but suffers from strong class bias. Decision Trees introduce non-linearity but are highly sensitive to overfitting. Random Forest provides the best overall balance between performance and robustness. Gradient Boosting offers additional flexibility but only limited gains in predictive power.

These results suggest that, while technical indicators contain some information about short-term market movements, the predictive signal remains weak. The next section compares model performances more directly and discusses the main modeling challenges, including overfitting, class imbalance, and the limits of short-term predictability.

6 Model Training and Technical Challenges

This section discusses the main technical aspects encountered during model training and the practical choices made throughout the modeling process.

6.1 Training Procedure and Computational Cost

All models were trained using a chronological train–validation–test split to respect the temporal structure of financial data and avoid look-ahead bias. This setup closely reflects real trading conditions.

From a computational perspective, training times remained reasonable. The full modeling pipeline, including feature construction, model training, and validation, required approximately three minutes on a standard personal computer. This runtime is acceptable for experimental and educational purposes.

It is worth noting that computational efficiency could be further improved. For instance, separating data preprocessing and feature construction into a precompiled dataset saved as an external file would avoid recomputing indicators at each execution. This approach would reduce repeated computation and slightly shorten overall runtime, but was not necessary for the scope of this project.

6.2 Model Complexity and Hyperparameter Tuning

Several models required careful control of complexity to prevent overfitting. Decision Trees were particularly sensitive to depth, with deeper trees achieving strong training performance but limited validation gains.

For ensemble models such as Random Forest and Gradient Boosting, hyperparameters including tree depth, number of estimators, learning rate, and minimum leaf size were adjusted using validation performance. All tuning was performed using a time-aware validation strategy rather than random cross-validation.

These results confirm that increasing model complexity does not automatically improve performance in short-term financial prediction tasks.

6.3 Evaluation Metrics and Class Balance

Model performance was evaluated using accuracy, precision, recall, F1-score, and ROC-AUC. Among these metrics, the F1-score was considered particularly informative, as it balances precision and recall in a context where prediction errors may have asymmetric financial consequences.

No class weighting was applied during model training. The class distribution was relatively balanced, and the focus of this project was placed on model comparison under consistent conditions rather than on optimizing performance through reweighting techniques.

7 Model Results and Comparison

This section presents the final model selection process and compares the performance of the best-performing models. At this stage, all model families have already been evaluated on the validation set. The objective here is to refine hyperparameters for the strongest candidates and to perform a single, unbiased evaluation on the test set.

To preserve methodological rigor, the modeling process follows a strict framework. Training is performed only on the training data. Hyperparameter selection relies exclusively on validation performance. The test set is used once, at the very end, to assess real out-of-sample performance. This procedure avoids overfitting to the test data and reflects realistic deployment conditions.

7.1 Final Model Selection and Hyperparameter Tuning

Based on validation results from the previous section, Random Forest and Gradient Boosting emerged as the most promising model families. A more detailed hyperparameter search was therefore conducted for these two models using a controlled grid search.

A time-aware validation strategy was implemented using a predefined split. Training observations were used to fit the models, while validation observations were used to evaluate each hyperparameter configuration. The F1-score was chosen as the optimization metric, as it balances precision and recall and is more informative than accuracy in this context.

The grid search results clearly favored the Random Forest model. The best Random Forest configuration used a relatively shallow depth and a large number of trees, which helps stabilize predictions while limiting overfitting. Gradient Boosting also showed reasonable performance, but consistently achieved lower validation F1-scores than Random Forest.

As a result, Random Forest was selected as the final model.

7.2 Final Evaluation on the Test Set

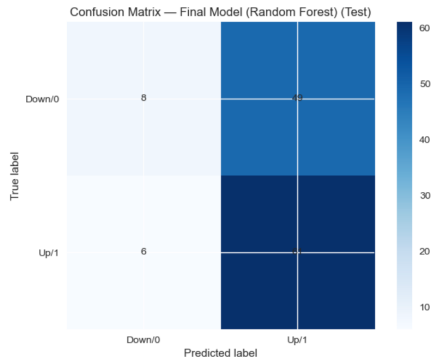
The selected Random Forest model was then evaluated on the test set, which had not been used at any previous stage of training or tuning.

On the test data, the model achieved an F1-score close to 0.69, with an accuracy slightly above 55%. The recall for positive returns is high, meaning that the model correctly identifies most upward market movements. However, recall for negative returns remains low, indicating that the model still struggles to detect downside movements.

This behavior is clearly visible in the confusion matrix. The model tends to predict the "up" class more frequently, which increases recall for positive returns but leads to missed negative signals. This asymmetry reflects the inherent difficulty of predicting short-term market declines and the weak signal-to-noise ratio in financial data.

The ROC-AUC on the test set remains close to 0.5, confirming that even the best-performing model only slightly improves upon random classification when evaluated in probabilistic terms.

Overall, these results confirm that while machine learning models can extract some information from technical indicators, short-term market direction remains extremely challenging to predict.



This confusion matrix shows the classification results of the final Random Forest model on the unseen test data.

The model correctly identifies most positive return days, which explains the high recall for the "up" class. This behavior is consistent with previous validation results and confirms that the model is biased toward predicting positive market movements.

However, the model struggles to correctly detect negative return days. A large number of true negative observations are misclassified as positive, leading to low recall for the "down" class. This asymmetry reflects the difficulty of predicting short-term market downturns and highlights the weak and noisy nature of downside signals.

Overall, the confusion matrix confirms that the model is better at capturing upward momentum than anticipating short-term declines, which is a common limitation in financial direction prediction tasks.

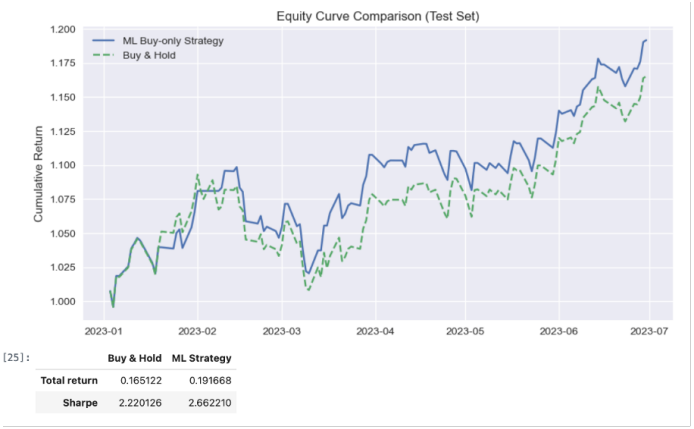
7.3 Economic Evaluation: Trading Strategy Comparison

Beyond classification metrics, the practical relevance of the model was evaluated using a simple trading strategy. A buy-only strategy was constructed, where the model invests in the market only when it predicts a positive next-day return. This strategy was compared to a passive Buy & Hold benchmark over the test period.

The equity curve comparison shows that the machine-learning-based strategy slightly outperforms the Buy & Hold benchmark in cumulative return. While both strategies follow a similar overall trend, the ML strategy avoids some negative return periods by staying out of the market when a negative signal is predicted.

In terms of risk-adjusted performance, the ML strategy also achieves a higher Sharpe ratio than Buy & Hold. This suggests that, despite modest classification accuracy, the model can still improve portfolio performance by reducing exposure during unfavorable periods.

However, it is important to emphasize that these results remain limited in magnitude. Transaction costs, slippage, and market impact are not considered in this simplified analysis. In real trading conditions, these factors could significantly reduce or eliminate the observed performance advantage.



[25]:

	Buy & Hold	ML Strategy
Total return	0.165122	0.191668
Sharpe	2.220126	2.662210

Figure 3: FINAL RESULT

This figure compares the cumulative performance of a buy-only strategy based on the machine learning model with a passive Buy & Hold strategy over the test period. The ML-based strategy achieves a higher final cumulative return than Buy & Hold. This improvement comes from staying invested mainly during periods where the model predicts positive market movements.

8 Interpretation, Limits, and Final Insights

This final section provides an overall interpretation of the results obtained in this project and discusses their main limitations. The objective is to connect the empirical findings to the initial research question and to place the results in a realistic financial and economic context.

The comparison of models highlights several important insights. First, ensemble methods clearly outperform simpler linear models. Logistic Regression, while easy to interpret, is unable to capture the complexity of short-term market dynamics and shows strong class bias. In contrast, Random Forest and Gradient Boosting benefit from their ability to model non-linear relationships and interactions between technical indicators. This suggests that non-linear patterns do exist in financial data and can be partially exploited by machine learning techniques.

At the same time, the results underline the importance of controlling model complexity. Decision Trees and ensemble models can easily overfit historical data if they are not properly constrained. Validation-based tuning shows that shallow trees and regularized ensemble models generalize better than overly complex ones. This confirms that, in financial applications, increasing model complexity does not automatically lead to better out-of-sample performance.

Despite these improvements, the overall predictive power of all models remains limited. Across all experiments, the highest classification accuracy achieved is around 61%. While this level of accuracy may appear modest, it must be interpreted in the proper financial context. Short-term stock market movements are highly noisy and influenced by new information that is difficult to anticipate.

In addition, the final trained model exhibits an asymmetric predictive behavior. The accuracy remains moderate, but the recall for positive returns is relatively high. This means that the model is able to correctly identify many upward market movements, while it struggles to detect downward movements. In other words, the model produces a large number of true positives but relatively few true negatives.

This asymmetric behavior has direct implications for strategy design. Since the model performs better at predicting market increases than decreases, a buy-only strategy was implemented. Under this framework, the model is used as a directional filter to decide when to enter the market, but it does not attempt to bet on price declines. This choice is consistent with both the model's strengths and realistic trading constraints, as short-selling introduces additional costs and risks.

From an economic perspective, the trading strategy based on the machine learning model slightly outperforms a passive Buy & Hold benchmark. The equity curve of the ML-based strategy appears smoother, with smaller drawdowns during some negative market phases. This improvement is reflected in a higher Sharpe ratio, indicating better risk-adjusted performance. These results show that even a model with limited classification accuracy can still add value when used as a selective entry mechanism rather than as a precise forecasting tool.

However, these results must be interpreted with caution. Transaction costs,

slippage, liquidity constraints, and market impact are not included in this analysis. In real trading conditions, these factors could significantly reduce or eliminate the observed performance advantage. Therefore, the strategy presented in this project should be viewed as an illustrative example rather than a directly deployable trading system.

In conclusion, this project confirms the central idea that machine learning can extract limited but non-zero information from non-linear technical indicators. At the same time, it clearly demonstrates that short-term stock market prediction remains fundamentally difficult. The results emphasize the importance of aligning modeling choices, evaluation metrics, and trading strategies with the realistic limits imposed by market efficiency.

9 Conclusion

This project studied whether short-term stock market returns can be predicted using machine learning models built on technical indicators and price-based features. Several supervised classification models were tested under a strict time-aware validation framework to ensure realistic evaluation.

The results show that non-linear and ensemble models perform better than simple linear approaches, which confirms the presence of weak non-linear patterns in financial data. However, overall predictive performance remains limited, with accuracy only slightly above random prediction. This outcome is consistent with market efficiency and highlights the difficulty of short-term market forecasting.

Despite these limits, the final model is able to correctly identify many upward market movements, which makes it useful as a directional filter. A simple buy-only strategy based on model predictions slightly outperforms a Buy & Hold benchmark in terms of cumulative return and risk-adjusted performance.

In conclusion, machine learning can extract limited but non-zero information from technical indicators, but short-term stock market prediction remains a challenging problem. Careful validation, realistic expectations, and appropriate strategy design are essential when applying machine learning in financial markets.

10 References and External Sources

The data used in this project were obtained from Yahoo Finance, which provides historical market data for major financial indices such as the S&P 500, Dow Jones Industrial Average, and CAC 40.

The implementation and analysis were carried out using the Python programming language and standard scientific libraries, including NumPy, pandas, scikit-learn, Matplotlib, and yfinance.

Technical indicators such as MACD, RS