electronics

MDPI

Article

Forecasting Significant Stock Market Price Changes Using Machine Learning: Extra Trees Classifier Leads

Antonio Pagliaro [1,2,3]

[1] INAF IASF Palermo, Via Ugo La Malfa, 153, 90146 Palermo, Italy; antonio.pagliaro@inaf.it
[2] Istituto Nazionale di Fisica Nucleare Sezione di Catania, Via Santa Sofia, 64, 95123 Catania, Italy
[3] ICSC—Centro Nazionale di Ricerca in HPC, Big Data e Quantum Computing, 40121 Bologna, Italy

Abstract: Predicting stock market fluctuations is a difficult task due to its intricate and ever-changing nature. To address this challenge, we propose an approach to minimize forecasting errors by utilizing a classification-based technique, which is a widely used set of algorithms in the field of machine learning. Our study focuses on the potential effectiveness of this approach in improving stock market predictions. Specifically, we introduce a new method to predict stock returns using an Extra Trees Classifier. Technical indicators are used as inputs to train our model while the target is the percentage difference between the closing price and the closing price after 10 trading days for 120 companies from various industries. The 10-day time frame strikes a good balance between accuracy and practicality for traders, avoiding the low accuracy of short time frames and the impracticality of longer ones. The Extra Trees Classifier algorithm is ideal for stock market predictions because of its ability to handle large data sets with a high number of input features and improve model robustness by reducing overfitting. Our results show that our Extra Trees Classifier model outperforms the more traditional Random Forest method, achieving an accuracy of 86.1%. These findings suggest that our model can effectively predict significant price changes in the stock market with high precision. Overall, our study provides valuable insights into the potential of classification-based techniques in enhancing stock market predictions.

Keywords: Random Forest; Extra Trees; machine learning; stock price forecasting

Citation: Pagliaro, A. Forecasting Significant Stock Market Price Changes Using Machine Learning: Extra Trees Classifier Leads. *Electronics* **2023**, *12*, 4551. https://doi.org/10.3390/electronics12214551

Academic Editor: Qian Yin

Received: 28 September 2023
Revised: 3 November 2023
Accepted: 4 November 2023
Published: 6 November 2023

1. Introduction

The role of the stock market in today's economy is invaluable. Investors are constantly seeking ways to anticipate the prices of their preferred stocks. With precise predictions, investors stand to gain substantial profits in the stock exchange. However, this task is not a simple one. The stock market is highly volatile and susceptible to various factors, including political events, economic conditions, trader attitudes, and behaviors, which can change abruptly. Additionally, due to its nonlinear and intricate nature, predicting stock prices remains a significant challenge.

The widely embraced Random Walk hypothesis [1] and the Efficient Market Hypothesis [2] have established the notion that forecasting the stock market is an impractical task. The Wisdom of Crowd hypothesis proposes that collective opinions may offer accurate predictions, but it has demonstrated limited efficacy in forecasting stock market returns. Additionally, the intricate nature of stock market prediction is further complicated by multiple variables and uncertainties that impact stock prices. However, despite these challenges, certain individuals and institutional investors have managed to surpass the market and achieve profitable outcomes [3].

The existence of predictable patterns in stock prices has been acknowledged by financial economists and statisticians, leading to controversial claims that investors can earn excess risk-adjusted returns [4]. However, statistical dependencies giving rise to momentum are exceedingly small, making it difficult for investors to realize excess returns. While

certain anomalies in stock returns exist, they are typically modeled within specific contexts and cannot be generalized. Trying to guess where stock markets are headed has been a highly sought-after and tough-to-solve problem for both investors and researchers. Experts in the field look at stock market trends with the help of sophisticated mathematics, computer science, economics, and other areas of knowledge. Notably, Technical Analysis, Time Series Forecasting, Machine Learning and Data Mining [5], and modeling and predicting volatility of stocks using differential equations [6] are methodologies used to predict stock price behavior.

In summary, stock market prediction is an important yet challenging task due to the complex and dynamic nature of financial markets. While various techniques have been applied to forecast stock prices and returns, there remains room for improvement in terms of predictive accuracy. This manuscript explores the potential of using a machine learning approach, which utilizes data mining techniques, to enhance stock market predictions. This aligns with the position advocated by Widom [7].

There has been a surge in researching and utilizing machine learning methods to predict stock prices in recent years, sparked by the influential work of Hellstrom and Holmstromm [5]. For a literature review see [8]. These include models of an artificial neural network, the support vector machines (SVMs) [9], the autoregressive integrated moving average (ARIMA) [10], the adaptive exponential smoothing and multiple regression [11]. More recently, Basak et al. [12] showed that Random Forests and gradient boosted decision trees are an improvement over previous predictions methods.

The main aim of this study is to utilize advanced machine learning models, some of which we already used in other fields of research [13,14], to develop an accurate prediction model for short-term stock market returns using an Extra Trees Classifier algorithm. Specifically, we predict if the return after 10 trading days (2 calendar weeks) will be significant by analyzing previous periods. The key objectives are: (1) to construct an Extra Trees Classifier model optimized for stock return predictions; (2) to evaluate its performance against benchmark models such as Random Forests; and (3) to assess its viability as an improved technique for stock market forecasting compared to traditional methods.

The key hypothesis is that an Extra Trees Classifier model can predict short-term stock returns more accurately compared to benchmark methods like Random Forests and traditional regression models.

Prior studies utilizing machine learning for stock prediction have focused predominantly on regression-based models. Classification models have been relatively underexplored despite their strengths in handling high-dimensional data and reducing overfitting. This presents a gap in evaluating classification algorithms like Extra Trees for stock market prediction. Extra Trees specifically overcomes drawbacks of Random Forests through faster training and robustness against noise, making it well-suited for complex financial data.

The paper is structured as follows: Section 2 covers methodology: training data preprocessing for classification and a description of all technical indicators evaluated. Section 3 provides an in-depth description of the dataset. Section 4 describes the sell/hold/buy strategy. Section 5 provides a description of models evaluated and estimation of features importance. The results are presented in Section 6 and include evaluation metrics and experimental results of simulated sells/buys following the signals from our model, and trading recommendations figures. A discussion follows. Additional information is in the Appendix A.

Our contributions include: (1) demonstrating the viability of a classification-based approach for stock forecasting compared to traditional regression models; (2) introducing a robust Extra Trees model optimized for stock market data; and (3) providing comparative evidence on the superior performance of Extra Trees over Random Forest algorithms for this task.

2. Methodology

The Extra Trees model is constructed using technical indicators derived from historical price data as input features and actual percentage returns over a 10-day period as targets. The model hyperparameters are tuned using grid search to optimize predictive performance. The dataset encompasses stocks from diverse sectors to improve generalizability.

Our method involves a comprehensive approach to data retrieval and preprocessing. To retrieve the data, we used a script that iterated through each symbol in a list of tickers, setting the start and end dates and downloading the data. We then calculated all the relevant technical indicators for each company before writing the data to a file for further analysis. The data were preprocessed to clean the data of any non-numeric values.

These indicators serve as a means of predicting future stock behavior and are used as features for training classifiers. This rigorous approach ensured that our data set was both accurate and reliable, providing a solid foundation for our subsequent analysis. The following section provides a detailed account of the techniques and technical indicators employed.

2.1. Training Data Processing

Data preparation is an important step in a machine learning method. It involves cleaning, transforming, and selecting the relevant data for the analysis. In this case, the features are technical parameters and the target is the market direction. These features are selected based on their relevance to the problem and their ability to provide meaningful insights. The target is used to train the machine learning model, so it must be defined clearly. When preparing the data for our model, the focus is on the target variable which represents the percentage difference in the stock price after a period of 10 trading days, or equivalently, 2 calendar weeks. In the realm of statistical analysis and machine learning, it is however necessary to categorize continuous variables into discrete groups to facilitate analysis. In this particular scenario, the target variable has been sorted into three classes utilizing pre-established bin boundaries, as explained in Section 4. Section 3 provides an in-depth description of the data set used.

2.2. Technical Indicators

In trees classifiers, feature selection refers to the process of determining which features, or input variables, are most important for making predictions. Technical indicators are mathematical calculations that are used to analyze and predict the behavior of financial markets, including stock markets. They are based on the historical price and/or volume data of a security and are used to identify trends, market conditions, and potential buy or sell opportunities.

The technical indicators used as features in our methods are the followings. It is worth noting that the technical indicators do not take into account any fundamental factors that might influence the price of an asset. Where not otherwise specified, period n is 14. For the computations, we used Pandas TA—A Technical Analysis Library in Python 3 [15].

2.2.1. Momentum Indicators

1. MACD. The Moving Average Convergence Divergence (MACD) is a technical indicator developed by Gerald Appel [16] that is used to measure the strength and direction of a trend.
   The formula to calculate the Moving Average Convergence Divergence (MACD) is:

$$\text{MACD(x,n,m)} = \text{EMA(x,n)} - \text{EMA(x,m)}$$

   where: EMA(x,n) = Exponential Moving Average of x with period n;
   n = number of periods for the fast moving average ;
   m = number of periods for the slow moving average .
   In our case, $m = 26$ and $n = 12$.

2. RSI. The Relative Strength Index (RSI) is a momentum oscillator that compares the magnitude of a stock's recent gains to the magnitude of its recent losses in an attempt to determine overbought and oversold conditions of an asset.
The RSI is calculated by taking the average gains of an asset over a certain number of periods, typically 14, and dividing it by the average losses over the same period. This results in a ratio, which is then converted into a number between 0 and 100. A reading above 70 is typically considered to indicate an overbought condition, while a reading below 30 is considered to indicate an oversold condition. The RSI being above 70 is termed an "overbought condition" because it signals the uptrend is exhausted and due for a reversal. While the asset may seem profitable in the short-term when RSI exceeds 70, the indicator is warning that upside momentum is unsustainable after such a sharp rise without pullbacks. The terminology "overbought" reflects the idea that the asset's price has been bought up too heavily, making a trend reversal likely ahead despite short-term profits. The formula to calculate the Relative Strength Index (RSI) is:

$$\text{RSI} = 100 - \frac{100}{(1 + \text{RS})}$$

where:
$\text{RS} = \frac{\text{Average gain of up periods}}{\text{Average loss of down periods}}$;
Average gain of up periods $= \frac{\sum_{i=1}^{n} U_i}{n}$;
Average loss of down periods $= \frac{\sum_{i=1}^{n} D_i}{n}$;
U_i is the gain of up periods;
D_i is the loss of down periods;
n is the number of periods.

3. TSI. The True Strength Index (TSI) is used to measure the strength of a trend in a financial market. TSI is similar to the Relative Strength Index (RSI) but it attempts to reduce the lag of the RSI by double-smoothing the data.
The TSI is calculated by first taking the rate of change of a security's closing price and then applying a double-smoothing process to the resulting value. The double-smoothing process involves calculating a simple moving average of the rate of change, and then calculating a second simple moving average of the first moving average.
Traders often use the TSI to identify potential overbought and oversold conditions in the market, by looking for bullish or bearish divergences between the TSI and the price action of the financial instrument. Bullish divergences occur when the TSI is making higher lows while the price is making lower lows, indicating that the price is likely to rise. Bearish divergences occur when the TSI is making lower highs while the price is making higher highs, indicating that the price is likely to fall. The TSI is calculated by taking the difference between the current closing price and the exponential moving average of the closing price, divided by the exponential moving average of the closing price, multiplied by 100. This results in an indicator that oscillates around the zero line, with values above zero indicating bullish momentum and values below zero indicating bearish momentum. The formula to calculate the TSI is:

$$\text{TSI}_i = 100 \times \frac{(Close_i - \text{EMA}_i)}{\text{EMA}_i}, \text{where: } \text{EMA}_i = \frac{Close_i + (n-1) \times \text{EMA}_{i-1}}{n}$$

where: TSI_i is the True Strength Index value for period i; $Close_i$ is the closing price for period i; EMA_i is the exponential moving average of the closing price for period i; n is the number of periods.

4. SLOPE TSI. The slope of the True Strength Index value is computed as numerical derivative of the True Strength Index. The slope of TSI is useful to identify the trend of the TSI, if it is increasing or decreasing.

5. RVGI. The Relative Vigor Index (RVGI) is used to measure the strength of a financial instrument's price action. The RVGI is calculated by first taking the difference between the current closing price and the previous closing price, and then calculating a moving average of these differences. The resulting value is then divided by the sum of the absolute differences between the current and previous closing prices, multiplied by 100.
   The formula to calculate the Relative Vigor Index (RVGI) is:
$$\text{RVGI}_i = \frac{U_i}{U_i + D_i} - \frac{U_{i-1}}{U_{i-1} + D_{i-1}}$$
   where: U_i is the Up period close; D_i is the Down period close; i is the current period; $i-1$ is the previous period.
   The RVGI indicator is a volatility-adjusted momentum oscillator that oscillates between -1 and 1, with values above 0 indicating bullish momentum and values below 0 indicating bearish momentum.
6. STC. The Schaff Trend Cycle (STC) is a combination of three different indicators: the cyclical component of the moving average (MACD), the rate of change (ROC), and a double-smoothed stochastic oscillator.
   The STC indicator uses the cyclical component of the MACD to identify the underlying trend in the market and the ROC and double-smoothed stochastic oscillator to identify short-term price movements. The indicator generates a signal when the short-term price movements diverge from the underlying trend, which can indicate a potential trend change.
   The formula for the STC is not publicly available as it is a proprietary indicator created by Doug Schaff. However, we used the adaption from Pandas TA library [15,17].
7. SLOPE STC. This feature is the slope of the exponential weighted moving average of the STC value.
8. Williams %R. The Williams %R [18], also known as the Williams Overbought/Oversold Index, is a momentum oscillator that measures overbought and oversold levels in the stock market. The formula for the Williams %R is given by:
$$\text{Williams \%R} = -100 * \frac{Highest\ \ High - Close}{Highest\ \ High - Lowest\ \ Low} \quad (1)$$
   where *Highest High* is the highest high for the period being analyzed, *Close* is the closing price, and *Lowest Low* is the lowest low for the period being analyzed. The Williams %R oscillates between 0 and -100, with values close to -100 indicating oversold conditions and values close to 0 indicating overbought conditions. In the following, Williams %R will be referred as WILLR.
9. CFO. The Chande Forecast Oscillator (CFO) is a momentum oscillator that measures the strength of price trends in the financial markets. It is calculated as the difference between the sum of recent gains and the sum of recent losses divided by the sum of all price changes over a given period. The CFO oscillates between positive and negative values, with positive values indicating a bullish market and negative values indicating a bearish market.
   The formula for the Chande Forecast Oscillator is as follows:
$$\text{CFO} = \frac{\sum_{i=1}^{n}(U_i - D_i)}{\sum_{i=1}^{n}(U_i + D_i)}$$
   where: U_i is the gain of up periods; D_i is the loss of down periods; n is the number of periods.

2.2.2. Overlap Indicators

1. SLOPE VWMA. The Volume-Weighted Moving Average (VWMA) is a technical indicator that combines the traditional moving average with volume data to give

more emphasis to periods of high trading activity. It is similar to a simple moving average but instead of equal weight to all prices, it gives more weight to periods with higher trading volumes.
The VWMA is calculated by taking the sum of the product of the closing price and the volume for each period, divided by the sum of the volume over the same period. This results in an average price that is more representative of the prices that were actually traded during the period.
The formula to calculate the Volume Weighted Moving Average (VWMA) is:

$$\text{VWMA} = \frac{\sum_{i=1}^{n} Close_i * Volume_i}{\sum_{i=1}^{n} Volume_i}$$

where: $Close_i$ is the closing price for period i; $Volume_i$ is the trading volume for period i; n is the number of periods.
Our feature is the gradient of the VWMA.

2. FWMA. The Fractal Weighted Moving Average (FWMA) is a variation of the simple moving average but it gives more weight to fractal patterns in the data. The FWMA is calculated by taking the sum of the product of the closing price and the weight of each fractal for each period, divided by the sum of the weights over the same period. The formula to calculate the Fractal Weighted Moving Average (FWMA) is:

$$\text{FWMA} = \frac{\sum_{i=1}^{n} Close_i * W_i}{\sum_{i=1}^{n} W_i}$$

where: $Close_i$ is the closing price for period i; W_i is the weight of fractal for period i; n is the number of periods.

2.2.3. Trend Indicators

1. ADX. The Average Directional Index (ADX) was calculated using a combination of three other indicators: the Plus Directional Indicator (+DI), the Minus Directional Indicator (−DI), and the Average True Range (ATR).
The +DI and −DI indicators are used to measure the strength of bullish and bearish movements, respectively, while the ATR is used to measure volatility.
They are calculated as follows: First, you need to calculate the True Range (TR) for each period. The True Range is the greatest of the following three values:

$$\text{TR} = \max(\text{high} - \text{low}, |\text{high} - \text{previous close}|, |\text{low} - \text{previous close}|)$$

The Plus Directional Movement measures the upward movement in price. It is calculated as follows:

$$+\text{DM} = \begin{cases} \text{ch} - \text{ph}, & \text{if } (\text{ch} - \text{ph}) > (\text{pl} - \text{cl}) \\ 0, & \text{otherwise} \end{cases}$$

where:
ch = current high;
ph = previous high;
cl = current low;
pl = previous low.
The Minus Directional Movement measures the downward movement in price. It is calculated as follows:

$$-\text{DM} = \begin{cases} \text{pl} - \text{cl}, & \text{if } (\text{pl} - \text{cl}) > (\text{ch} - \text{ph}) \\ 0, & \text{otherwise} \end{cases}$$

The Average True Range (ATR) measures market volatility and is calculated as an average of the TR values over a specified period.

Now we can calculate the Plus Directional Indicator (+DI) and Minus Directional Indicator (−DI) as follows:

$$+\text{DI} = \frac{\text{14-period EMA of +DM}}{\text{ATR}}$$

$$-\text{DI} = \frac{\text{14-period EMA of } -\text{DM}}{\text{ATR}}$$

Here, EMA stands for Exponential Moving Average, and the "14-period" indicates that you typically use a 14-period EMA for these calculations. You calculate EMA by giving more weight to recent data points, which helps to smooth out the values and make the indicators more responsive to recent price movements.
The ADX is then calculated by taking the absolute difference between the +DI and −DI and dividing it by the sum of the +DI and -DI, multiplied by the ATR.
The formula to calculate the Average Directional Index (ADX) is:

$$\text{ADX} = \frac{|+\text{DI} - -\text{DI}|}{+\text{DI} + -\text{DI}} \times \text{ATR}$$

where: +DI is the Plus Directional Indicator; −DI is the Minus Directional Indicator; ATR is the Average True Range.
The ADX is a measure of the overall trend strength, combining the strength of both bullish and bearish movements while taking market volatility into account using the ATR. It helps traders and analysts identify the strength of a trend and whether it is worth trading.

2. AROON. The Aroon is used to measure the strength of a trend and the likelihood of its continuation, and consists of two lines, the Aroon Up line and the Aroon Down line. The Aroon Up line measures the strength of the uptrend, while the Aroon Down line measures the strength of the downtrend.
The Aroon Up line is calculated by taking the number of periods since the highest high divided by the total number of periods. The Aroon Down line is calculated by taking the number of periods since the lowest low divided by the total number of periods.
Values for Aroon Up and Aroon Down oscillate between 0 and 100, with readings near 100 indicating a strong trend in the direction of the oscillator and readings near 0 indicating a weak trend or no trend at all.
Aroon Up and Aroon Down indicators can be used in combination to identify potential trend changes. An increasing Aroon Up and a decreasing Aroon Down indicate a strong uptrend, while a decreasing Aroon Up and an increasing Aroon Down suggest a strong downtrend. A weak trend or no trend at all is indicated when both Aroon Up and Aroon Down decrease.
Additionally, a buy signal is generated when Aroon Up crosses Aroon Down from below, and a sell signal is generated when Aroon Down crosses Aroon Up from above.
The formula for calculating the Aroon Up line is:

$$\text{Aroon Up} = \frac{(\textit{Days Since Highest High})}{N} \times 100$$

where:
N is the number of periods;
$\textit{Days Since Highest High}$ = N − Days Since N − Period High
and Days Since N − Period High is the number of periods that have passed since the highest high within the last N periods. It is calculated by counting the number of periods between the current period and the period when the highest high occurred. In this case, N is set to 14.
The formula for calculating the Aroon Down line is:

$$\text{Aroon Down} = \frac{(Days\ Since\ Lowest\ Low)}{N} \times 100$$

where:
N is the number of periods;
$Days\ Since\ Lowest\ Low$ = N − Days Since N − Period Low
and Days Since N -Period Low is the number of periods since the lowest low. Our feature is computed as:

$$\text{Aroon} = \text{Aroon Up} - \text{Aroon Down}$$

2.2.4. Volatility Indicators

1. Bollinger Bands. Bollinger Bands are used to measure the volatility of a financial instrument. The indicator consists of three lines: the simple moving average line, which is the middle band; and an upper and lower band. The upper band is plotted two standard deviations above the simple moving average, while the lower band is plotted two standard deviations below the simple moving average.
   Bollinger Bands are often used to identify potential overbought and oversold conditions in the market. When the price of a financial instrument moves above the upper band, it is considered overbought, and when it moves below the lower band, it is considered oversold. Traders can also use Bollinger Bands to identify potential breakouts by looking for price action to break above or below the upper or lower bands.
   Traders also use Bollinger Bands as a volatility indicator, by using the width between the upper and lower bands. The wider the bands, the higher the volatility, and the narrower the bands, the lower the volatility.
   The formula to calculate the three Bollinger Bands are:
   Middle Band = SMA(n)
   Upper Band = SMA$(n) + k \cdot$ standard deviation(n)
   Lower Band = SMA$(n) - k \cdot$ standard deviation(n)
   where: SMA(n) is the Simple Moving Average with period n; standard deviation(n) is the standard deviation of x with period n; n is the number of periods for the moving average. In our case $n = 14$; k is the number of standard deviations from the moving average to plot the upper and lower bands. In our case $k = 2$.
   In the following, medium, upper, and lower Bollinger bands will be referred to as BOLL M, BOLL U, and BOLL L.
2. RVI. The Relative Volatility Index (RVI) is used to measure the volatility of an asset relative to its own recent price history. The RVI formula is defined as follows:
   RVI = EMA$(|Close_t - Close_{t-1}|)$
   where EMA is the Exponential Moving Average, $Close_t$ is the closing price at time t, and $Close_{t-1}$ is the closing price at time $t - 1$. The RVI ranges from 0 to 100 and is typically used to identify overbought and oversold conditions in the market.
3. Price from Donchian. The Donchian channel is a moving average of the highest high and the lowest low prices over a certain period of time and is typically plotted on a chart as three lines: the upper line represents the highest high price over the specified time period, the middle line represents the average of the highest high over the specified time period, and the lower line represents the lowest low price over the specified time period.
   The price of the asset oscillates between these two bands, and when the price breaks above the upper band, it is considered to be in an uptrend, and when it breaks below the lower band, it is considered to be in a downtrend.
   The two bands are calculated as follows:
   Upper band = $\max_{i=1}^{n} High_i$
   Lower band = $\min_{i=1}^{n} Low_i$
   where: $High_i$ is the highest price for period i; Low_i is the lowest price for period i; n is the number of periods. In our case: $n = 28$.

Our feature is the distance of the closing price from the mean of the Donchian channel and is computed as follow:

$$\mathrm{DC}_t = \frac{Closing\ \ Price_t - Mean\ \ (Upper\ \ Band_t, Lower\ \ Band_t)}{Mean\ \ (Upper\ \ Band_t, Lower\ \ Band_t)}$$

where: $Closing\ \ Price_t$ is the closing price at time t; $Upper\ \ Band_t$ is the upper band at time t; $Lower\ \ Band_t$ is the lower band at time t; $Mean(x, y)$ is the mean of x and y. In the following, this feature will be referred as PF DONCHIAN.

4. Slope of Price from Donchian. This feature is the slope of the previous indicator. In the following, this feature will be referred to as SLOPE PFD.

2.2.5. Volume Indicators

1. A/D. Accumulation/Distribution (A/D) measures the buying and selling pressure of a financial instrument. The A/D indicator is calculated by taking the difference between the current close and the previous close, multiplied by the volume. If the current close is above the previous close, the value is positive, indicating that money is flowing into the stock.
   The formula to calculate the Accumulation/Distribution is:
   $\mathrm{AD}_i = \mathrm{AD}_{i-1} + (Close_i - Close_{i-1}) \times Volume_i$
   where: AD_i is the Accumulation/Distribution value for period i; $Close_i$ is the closing price for period i; $Close_{i-1}$ is the closing price for period $i-1$; $Volume_i$ is the trading volume for period i.
   This formula calculates the A/D by taking the difference between the current close and the previous close, multiplied by the volume, and adding the result to the previous A/D value. If the current close is above the previous close, the value is positive, indicating that money is flowing into the stock, and therefore it is considered a buying pressure. If the current close is below the previous close, the value is negative, indicating that money is flowing out of the stock, and therefore it is considered a selling pressure.
2. SLOPE AD. This feature is the slope of the previous indicator.
3. CMF. The Chaikin Money Flow (CMF) is based on the Accumulation/Distribution line and is calculated by taking the difference between the high and low prices for each period, multiplied by the volume, and dividing the sum of these values by the sum of the volume over the same period. This results in an indicator that oscillates around the zero line, with values above zero indicating buying pressure and values below zero indicating selling pressure.
   The formula to calculate the Chaikin Money Flow (CMF) is:

$$\mathrm{CMF} = \frac{\sum_{i=1}^{n}[(Close_i - Low_i) - (High_i - Close_i)] * Volume_i}{\sum_{i=1}^{n} Volume_i}$$

   where: $Close_i$ is the closing price for period i; Low_i is the lowest price for period i; $High_i$ is the highest price for period i; $Volume_i$ is the trading volume for period i; n is the number of periods. This formula calculates the CMF by taking the difference between the high and low prices for each period, multiplied by the volume, and dividing the sum of these values by the sum of the volume over the same period.

2.2.6. More Indicators

1. SLOPE A50. The feature SLOPE A50 is computed applying the Exponentially Weighted Moving Average to the closing prices using a window of 50 days, and calculating the gradient.
2. SLOPE A23. Same as the previous feature, with a window of 23 days.

2.3. Exponential Smoothing

For the computation of the features STC, SLOPE STC, SLOPE A50, and SLOPE A23, the time series historical stock data was first exponentially smoothed. Exponential smoothing is a time series method that uses a weighted average of past observations. The weights decrease exponentially as the observations get older, hence the name "exponential smoothing".

There are several different types of exponential smoothing, including:

- Simple exponential smoothing: This method uses a single smoothing factor to give more weight to recent observations and less weight to older observations.
- Holt's linear exponential smoothing: This method adds a trend component to simple exponential smoothing, allowing for the prediction of both level and trend in the data.
- Holt–Winters exponential smoothing: This method adds a seasonal component to Holt's linear exponential smoothing, allowing for the prediction of level, trend, and seasonality in the data.

The choice of which type of exponential smoothing to use depends on the characteristics of the time series data. Simple exponential smoothing is best for data with no clear trend or seasonality. This is our case. Holt's linear exponential smoothing is best for data with a clear trend but no seasonality, and Holt–Winters exponential smoothing is best for data with both a clear trend and seasonality.

The determination of the smoothing factor (alpha) for exponential smoothing involves minimizing the sum of squared errors between predicted and actual values.

For our scenario, the simple exponential smoothing of a series Y can be calculated recursively using the following formula:

$$S_t = \alpha Y_t + (1 - \alpha) S_{t-1} \quad \text{for} \quad t \geq 1$$

Here, the smoothing factor, denoted by $0 < \alpha < 1$, is used to reduce the level of smoothing. A larger value of α results in less smoothing, whereas at $\alpha = 1$, the smoothed statistic becomes equal to the observed value. Smoothing eliminates random fluctuations from historical data, thereby making it easier to identify the long-term trend of a stock price. We use $\alpha = 0.9$.

3. Data Set

The analysis presented in this paper aims to predict the percentage difference between the closing price and the closing price after 10 trading days (2 calendar weeks). The data used for the analysis includes 120 companies (listed in the Appendix A), covering the time period from 1 January 1995 to the end of 2022. All available data for these companies were used, from day 1 (or from their IPO, if they were listed after 1 January 1995) until the end of the aforementioned time period, or until the date in which they were delisted. The sample of companies was chosen randomly, without any strict criteria regarding their background or economic impact. The companies in the sample represent a diverse range of industries, including software, electronics, and pharmaceutical companies, among others.

The raw data used in the analysis includes the date of entry, closing price, and volume, among other variables. The target variable for the analysis is the percentage difference between the closing price and the closing price after 10 trading days (2 calendar weeks).

The selection of a suitable time frame for evaluating trading strategies is a critical component of technical analysis. Short time frames, such as 3 or 5 days, may not allow for a reliable assessment of a strategy's accuracy due to the high unpredictability of short-term price movements (in Basak et al. [12], Random Forest models have been evaluated for various trading windows, including 3, 5, 10, 15, 30, 60, and 90 days, and have shown an increase in accuracy with longer windows. Interestingly, accuracies for windows less than 10 days were only slightly greater than 50%). This is because short-term price movements can be erratic and unpredictable, resulting in low accuracy levels when using these time frames. On the other hand, longer time frames can provide more significant trends, but they may not be practical for traders who want to make multiple trades in a month.

Thus, the choice of a 10-day time frame can be considered a good compromise between accuracy and practicality. This time frame enables traders to evaluate the performance of their strategy within a reasonable time frame that allows for more predictable price movements while still being practical practical for traders who wish to make frequent trades. Additionally, this time frame aligns with the two-week cycle of many economic indicators and news events that can impact market movements.

It is worth noting that the choice of a time frame for technical analysis can vary depending on the specific asset being analyzed and the trading strategy being employed.

The code used to retrieve the data involved iterating through each symbol in the list of tickers, setting the start and end dates, and downloading the data. The downloaded data were then preprocessed and cleaned of any non-numeric values. Next, all the technical indicators listed before for each company were computed and written to a file for further analysis.

All feature values were continuous, with no categorical or ordinal variables included. Non-linear trends were observed among most features, making tree-based classifiers an attractive option for analysis.

4. Strategy

In statistical analysis and machine learning, it is common to bin continuous variables into discrete categories for analysis. In this case, the target variable has been binned into three classes using predetermined bin edges.

In order to ensure a balanced distribution of data among the bins, the bin edges were initially chosen based on a balance of preferred closing price percentage differences and a similar number of data in each bin. This approach helps to avoid bias in the analysis by ensuring that each bin contains a representative sample of the data. Once the bins were established, they were pruned so that each bin contained an equal number of samples. This further improves the fairness of the analysis by ensuring that each bin carries the same weight in the analysis. The process is known as stratified sampling and is particularly useful when dealing with imbalanced datasets where the number of samples in each class is not equal. By using stratified sampling, we can ensure that the resulting model is not biased towards one particular class and is better able to accurately predict outcomes for each class.

- The first bin includes all values less than -0.03, which corresponds to a loss in closing price after 10 trading days of more than 3%. In this bin, the resulting strategy would be to sell.
- The second bin includes all values between -0.03 and 0.04, which corresponds to a price change within the range of -3% to +4%. In this bin, the resulting strategy would be to hold.
- The third bin includes all values greater than 0.04, which corresponds to a gain in closing price after 10 trading days of more than +4%. In this bin, the resulting strategy would be to buy.

5. Prediction Models

5.1. Random Forest

A Random Forest is an ensemble machine learning method for classification and regression. It consists of a collection of decision trees, where each tree is trained on a random subset of the data. During prediction, the Random Forest aggregates the predictions of each individual tree to arrive at the final output. The idea behind this is that by training multiple trees on different subsets of the data, the overall performance of the model can be improved by reducing overfitting and increasing the robustness of the predictions. Additionally, Random Forests also provide a measure of feature importance which can be used for feature selection.

The concept of a Random Forest was first introduced in the early 2000s by Leo Breiman. In his paper published in 2001 [19], he described the method as a combination of bagging

(bootstrap aggregating) and random subspace method. In bagging, multiple models are trained on different subsets of the training data, while in the random subspace method, only a random subset of the features are considered for each split in the decision tree.

The idea behind Random Forests was to improve the performance of decision trees, which often suffer from overfitting when trained on large datasets. By averaging the predictions of multiple decision trees, Random Forests are able to reduce overfitting and improve the robustness of the predictions.

5.2. Extra Trees

Extra Trees Classifier [20] is an ensemble machine learning algorithm used for classification tasks. It belongs to the Random Forest family of algorithms, which builds multiple decision trees and combines their predictions to form a final output. The Extra Trees Classifier differs from traditional Random Forest in that it uses random thresholds for each feature, instead of selecting the best split, to form each decision tree. This randomization process results in a more diverse set of trees and therefore a more robust final prediction.

5.3. A Comparison of Models

We built, tuned, and compared thirty machine learning classifier models in order to identify the best performing ones for our specific cases.

It is worth noting that training and testing a set of models using the full dataset or even larger subsets can significantly increase computational costs and time requirements. Therefore, we opted for a smaller subset of data to facilitate more efficient experimentation and analysis. Therefore, although our dataset comprises around 500,000 samples, for the purposes of comparison we opted to use a smaller training set consisting of 80,000 samples and a testing set of 20,000 samples. This reduced set includes all 24 features.

More relevant models considered were:

- Bagging Classifier (Bootstrapped Aggregating) is an ensemble machine learning technique that combines the predictions of multiple models to improve the stability and accuracy of the prediction. The method involves training multiple models on randomly sampled subsets of the training data, and then averaging the predictions of each model.
- Decision Tree Classifier is a supervised learning algorithm used for classification problems. It models decisions and decision making by learning from the data to construct a tree-like model of decisions and their possible consequences. The algorithm works by recursively splitting the data into subsets based on the feature that results in the largest information gain and assigning a class label to each leaf node in the tree. The final class label of a sample is determined by traversing the tree from the root to a leaf node. Decision trees are simple to understand, interpret, and visualize and can handle both categorical and numerical data.
- NuSVC is a Support Vector Machine (SVM) classifier which uses a nu-parameter to control the number of support vectors. It works by finding the optimal hyperplane that separates the data points of different classes with the maximum margin. The nu-parameter controls the trade-off between margin size and number of support vectors. NuSVC is useful when dealing with large datasets as it uses a subset of training points, known as support vectors, in the decision function.
- XGB Classifier is an implementation of gradient boosting algorithm for classification problems. It builds a sequence of decision trees to make the final prediction. The algorithm works by iteratively adding new trees to the model, with each tree trained to correct the errors of the previous ones. The XGB Classifier also includes regularization techniques, such as L1 and L2 regularization, to prevent overfitting.
- KNeighbors Classifier is a type of instance-based learning algorithm, which uses a non-parametric method to classify new data points based on the similarity of their features to those of the training data. It works by assigning a class label to a new data point based on the class labels of its k-nearest neighbors in the training set. The value

of k is chosen by the user and determines the number of neighbors to consider when making a prediction. KNeighbors Classifier is simple to implement and works well for small datasets with few dimensions.

- LGBM Classifier is a gradient boosting framework that uses tree-based learning algorithms. It stands for Light Gradient Boosting Machine, and it is a scalable and efficient implementation of gradient boosting specifically designed to handle large datasets.
- Quadratic Discriminant Analysis (QDA) is a linear classification method that assumes a Gaussian distribution of the features within each class and estimates the class covariance matrices. QDA uses these covariance matrices to calculate the discriminant function that separates the classes. Unlike Linear Discriminant Analysis (LDA), QDA does not assume equal covariance between the classes and therefore provides a more flexible boundary between the classes. It is effective in cases where the class covariance matrices are different and the classes are well-separated.

Results are shown in Table 1 and show that the best performing model for our data is the Extra Trees Classifier.

Table 1. A comparison of machine learning methods.

Model	Accuracy
Extra Trees Classifier	0.75
Random Forest Classifier	0.73
Bagging Classifier	0.67
Decision Tree Classifier	0.63
NuSVC	0.57
XGB Classifier	0.55
KNeighbors Classifier	0.54
LGBM Classifier	0.52
Quadratic Discriminant Analysis	0.44

5.4. Random Forest vs. Extra Trees

Table 1 provides a comprehensive overview of various machine learning methods. Random Forest and Extra Trees emerge as the best-performing candidates. Their high accuracy scores make them standout models, and therefore, they warrant special attention and further scrutiny.

Extra Trees and Random Forest are both ensemble learning methods that combine multiple decision trees to improve predictive performance. However, there are some key differences between the two algorithms.

Extra Trees is computationally faster. Extra Trees randomly selects features and thresholds for each split, whereas Random Forest selects the best feature and threshold. This means that Extra Trees requires less computation and can train more quickly.

Another advantage of Extra Trees is that it can reduce overfitting. In Random Forest, each tree is built using a bootstrap sample of the training data, which can result in correlated trees that overfit to the training data. Extra Trees, on the other hand, uses a random subset of the training data and random splits, which can reduce overfitting and improve generalization performance.

Extra Trees is a better choice for a large dataset with many features and needs to reduce computation time and overfitting.

5.5. Feature Importances

The mean decrease in impurity is used to calculate feature importance. This method assesses the reduction in the impurity criterion achieved from all the splits made by the trees based on a particular feature, and determines which features are more relevant. Tree-based algorithms have an in-built feature importance mechanism. However, tree-based models tend to overestimate the significance of continuous numerical features, as these features provide more opportunities for the models to split the data in half. To overcome

this, we used the permutation feature importance method. We trained the model on the training set and obtained the model score on the test set, which was used as the baseline. Subsequently, we shuffled one feature at a time on the test set and fed it to the model to obtain a new score. If the feature that was shuffled is significant, the model's performance should degrade significantly and the score should drop drastically. On the other hand, if the feature is not important, the model's performance should remain unaffected.

Our results are summarized in Table 2. A random feature is added for control.

Table 2. Extra Trees features importances.

Features	Importance
PF DONCHIAN	0.0523
BOLL L	0.0489
BOLL M	0.0464
A/D	0.0459
FWMA	0.0457
BOLL U	0.0454
ADX	0.0451
MACD	0.0434
CFO	0.0415
SLOPE A50	0.0413
RVGI	0.0409
CMF	0.0400
TSI	0.0397
AROON	0.0397
SLOPE A23	0.0388
SLOPE PFD	0.0383
STC	0.0379
RVI	0.0369
RSI	0.0364
SLOPE VWMA	0.0364
WILLR	0.0363
SLOPE STC	0.0349
SLOPE TSI	0.0342
SLOPE AD	0.0327
RANDOM	0.0028

5.6. Another Features Importance Estimation

We explored another method to investigate the accuracy of Extra Trees Classifier in predicting the target variable using a subset of features, after pruning the variables based on binning and downsampling. The study used a range of feature combinations with at least 10 and at most 24 features, and evaluated each model using accuracy scores.

It is important to note that this method carries a high computational cost as it required iterations on approximately 14 million models. As a result, we reduced the data sample size to 10,000. We analyzed the occurrence of the number of features and found that 15 features were the most frequently occurring, indicating that this number may provide an optimal balance between model complexity and performance. Please refer to the results of Table 3 for more details.

Table 3. This table summarizes the occurrences of the number of features in the top 100 performing models.

Number of Features	Occurrences
15	30
16	11
23	9
20	8
17	7

Our analysis revealed that the most frequently occurring number of features was 15, suggesting that this number may provide a good balance between model complexity and performance.

Furthermore, we identified the most commonly occurring features in the top performing models, which are good candidates as best features for our model. These features are CFO, ADX, PF DONCHIAN, TSI, MACD, BOLL L, BOLL M, BOLL U, A/D, FWMA, SLOPE PFD, SLOPE TSC, SLOPE AD, SLOPE A50, and SLOPE A23.

5.7. Selected Features

A total of 24 features were used for the evaluation and comparison based on the two feature importance methods mentioned before. The first method used permutation feature importance. The second method used Extra Trees Classifier and identified a set of 15 features that were commonly occurring in the top performing models.

Therefore, it is reasonable to consider the intersection of the important features from both methods and use a number of 15 features, as suggested by the second method. After more tests on intersections, we have selected the following 15 features for our model: CFO, ADX, SLOPE STC, SLOPE PFD, PF DONCHIAN, TSI, SLOPE A50, SLOPE A23, MACD, BOLL L, BOLL M, BOLL U, A/D, FWMA, and SLOPE AD.

We chose these features because they consistently appeared among the most important features in both methods of feature selection. Moreover, these features have been shown to have a high degree of predictive power for our target variable based on prior research and domain knowledge. Additionally, we believe that these features provide a good balance between model complexity and performance, and should therefore allow us to build a robust and accurate model.

5.8. Hyperparameter Tuning

Hyperparameter tuning was conducted through a randomized grid search, which leverages both the "fit" and "score" methods, focusing on the following hyperparameters:

- *n estimators* number of decision trees in the Random Forest;
- *max features* maximum number of features that are considered at each split in the decision tree;
- *max depth* maximum depth of the decision trees in the Random Forest;
- *min samples split* minimum number of samples required to split an internal node in the decision tree;
- *minimum samples leaf* minimum number of samples required to be at a leaf node in the decision tree;
- *bootstrap*: Bootstrapping in Machine Learning involves creating multiple subsets of training data by randomly sampling with replacement from the original dataset. This technique is used to train multiple decision trees, each on a different subset of the training data. The "bootstrap" hyperparameter is a Boolean value that determines whether bootstrapping is applied during tree construction. When set to "True", bootstrapping is used to create diverse training subsets for each decision tree, aiding in reducing overfitting and enhancing generalization. Conversely, when set to "False", each decision tree is trained on the entire dataset.

Our best results are as follows:

- *n estimators* = 200;
- *max features* = auto; the algorithm automatically chooses the appropriate number of features to consider, based on a square root of the total number of features available in the dataset;
- *max depth* = 80;
- *min samples split* = 2;
- *min samples leaf* = 1;
- *bootstrap* = False.

6. Results

6.1. Evaluation Metrics

Accuracy, precision, recall, specificity, balanced accuracy, and F1 score are all evaluation metrics used to evaluate the performance of a classifier, such as a Random Forest or a Extra Trees Classifier.

Accuracy is the proportion of correctly classified instances out of the total number of instances. It is defined as (True Positives + True Negatives)/Total. Precision is the proportion of true positive predictions out of all positive predictions. It is defined as True Positives/(True Positives + False Positives). Recall (also known as sensitivity) is the proportion of true positive predictions out of all actual positive instances. It is defined as True Positives/(True Positives + False Negatives). Specificity is the proportion of true negatives out of all actual negatives instances. It is defined as True negatives/(True negatives + False positives). Balanced accuracy is the arithmetic mean of sensitivity and specificity, which is useful when the classes are imbalanced. It is defined as (True Positives/(True Positives + False Negatives) + True negatives/(True negatives + False positives))/2. The F1 score is the harmonic mean of precision and recall, which is useful when the classes are imbalanced or when both precision and recall are important. It is defined as 2 ∗ (Precision ∗ Recall)/(Precision + Recall).

Here are the formulas:

Accuracy:

$$\frac{TP + TN}{Total}$$

Precision:

$$\frac{TP}{TP + FP}$$

Recall (Sensitivity):

$$\frac{TP}{TP + FN}$$

Specificity:

$$\frac{TN}{TN + FP}$$

Balanced accuracy:

$$\frac{1}{2} \cdot \left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right)$$

F1 score:

$$2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

where: *TP* is True Positives, *TN* is True Negatives, *FP* is False Positive, and *FN* is False Negative.

The dimensions of the training and testing datasets are given as follows: training features shape is 409,025 for 15 features, training labels shape is 409,025, testing features shape is 72,181 for 15 features, and testing labels shape is 72,181. Our results for the the Extra Trees Classifier model are summarized in Table 4. Results for the selected features importances are summarized in Table 5.

The model achieved a training score of 0.9989 and a testing score of 0.8608. The accuracy, precision, recall, balanced accuracy, and F1 score of the model were all around 0.86. These results indicate that the model was highly accurate in predicting the direction of stock market prices. It is worth noting that Basak [12] reported results for a Random Forest model, which showed an accuracy of around 80% or less (depending on the stock) for a 10-day trading window. In addition, it should be mentioned that while the Random Forest model in Basak [12] has only two (buy and sell), the Extra Trees Classifier model has three values for the target (buy, hold, and sell) and allows for a more nuanced prediction of stock market prices. Therefore, the Extra Trees Classifier model appears to outperform the Random Forest model.

Table 4. Performance Metrics of the Extra Trees Classifier Model.

Results	
Training score	0.9989
Testing score	0.8608
Accuracy (test data set)	0.8608
Precision (test data set)	0.8614
Recall (test data set)	0.8608
Balanced Accuracy (test data set)	0.8608
F1 score (test data set)	0.8610
Specificity (test data set)	0.8403

Table 5. Features importances. Please note that the ranking of a feature can be strongly influenced by how it interacts with other features. In the full feature set of Table 2, "PF DONCHIAN" has favorable interactions that boost its importance or performance. In this table only a subset of features is considered and the absence of specific interacting features diminishes its ranking.

Features Importances	
BOLL L	0.0748
BOLL M	0.0718
ADX	0.0717
BOLL U	0.0716
FWMA	0.0712
TSI	0.0710
MACD	0.0708
PF DONCHIAN	0.0682
A/D	0.0668
SLOPE50	0.0651
CFO	0.0624
SLOPE23	0.0624
SLOPE STC	0.0611
SLOPE PFD	0.0574
SLOPE AD	0.0536

6.2. Experimental Results

Simulations were conducted to evaluate the performance of our stock trading model. The simulations involved investing in each of 15 stocks, namely: ALB, CAT, CSCO, DHI, DHR, DPZ, ENPH, IDXX, LRCX, MCD, MU, PG, TGT, TWTR, and XOM. These stocks were not used during the training and testing of the model, and therefore were independent of it.

At the beginning of the simulations on 18 January 2022, $10,000 was invested in each stock. The simulations ran until 21 February 2023. During this period the trading signals generated by the model were used to buy and sell the stocks. The value of the portfolio at the end of the simulations was calculated in two ways. The first method calculated the portfolio value based on the trading signals generated by the model, while the second method calculated the portfolio value assuming that the stocks were held without any trading activity.

The results of the simulations (see Table 6) showed that the model generated positive returns for twelve stocks and negative returns for the remaining four. Holding gives positive returns only in six cases out of fifteen. Moreover, for all the stocks considered, the returns from the model were better than the ones from the hold strategy: the percentage difference between the portfolio value based on the trading signals generated by the model and the portfolio value assuming that the stocks were held without any trading activity was positive for all the stocks.

The total investment was $150,000. At the end of the simulations, the total portfolio value based on the trading signals generated by the model was $178,638.89, while the total portfolio value assuming that the stocks were held without any trading activity was $157,116.46.

It is worth noting that during this period, the market was turbulent, with the Federal Reserve increasing interest rates. As a result, the S&P 500 index experienced a decline of approximately 12.7%, falling from 4577 to 3997.

The results of the simulations suggest that the trading model has the potential to generate positive returns for some stocks, but not for all stocks. The percentage difference between the portfolio value based on the trading signals generated by the model and the portfolio value assuming that the stocks were held without any trading activity provides an indication of the effectiveness of the trading signals.

Table 6. Stock returns and comparison of trading strategy with holding strategy for the period 18 January 2022 to 21 February 2023.

Stock	Invested	Asset Trading	Asset Hold	Trading Strategy Return	Hold Return	% Difference
ALB	$10,000	$10,846.27	$10,692.00	+8.46%	+6.92%	+1.54%
CAT	$10,000	$10,681.18	$10,471.57	+6.81%	+4.72%	+2.09%
CSCO	$10,000	$9102.07	$8319.10	-8.98%	−16.81%	+7.83%
DHI	$10,000	$10,244.50	$9661.74	+2.45%	−3.38%	+5.83%
DHR	$10,000	$11,303.50	$8695.91	+13.04%	−13.04%	+26.08%
DPZ	$10,000	$10,399.20	$7508.25	+3.99%	−24.92%	+28.91%
ENPH	$10,000	$14,904.29	$14,835.93	+49.04%	+48.36%	+0.68%
IDXX	$10,000	$9806.33	$9228.05	−1.94%	−7.72%	+5.02%
LRCX	$10,000	$12,559.28	$7043.93	+25.59%	−29.56%	+55.15%
MCD	$10,000	$10,959.96	$10,470.60	+9.60%	+4.71%	+4.89%
MU	$10,000	$8490.96	$6202.22	−15.09%	−37.98%	+22.89%
PG	$10,000	$11,616.68	$8926.82	+16.17%	−10.73%	+26.90%
TGT	$10,000	$10,158.78	$7555.11	+1.59%	−24.45%	+26.04%
TWTR [1]	$10,000	$16,841.23	$14,396.78	+68.41%	+43.97%	+24.44%
XOM	$10,000	$15,445.01	$15,212.10	+54.45%	+52.12%	+2.15%
Total	$150,000	$178,638.89	$157,116.46	+19.09%	+4.74%	+14.35%

[1] TWTR stock was delisted from the NYSE on 8 November 2022 after Elon Musk bought all the company's outstanding shares for $54.20 per share. Therefore, both models were forced to sell on the date for that price.

6.3. Trading Recommendations

Figures 1 and 2 display trading recommendations generated by our model for a selection of equities not used during the training and testing of the model, and therefore independent of it. The model's suggested trading decisions are indicated by colored dots. Specifically, red dots denote a forecasted drop in prices, while green dots denote a forecasted rise in prices, both occurring within a 10 day interval. The model's predictions serve as a guide for the recommendation of a purchase or sell decision. The graph indicates that the model advocates for purchasing when an increase in prices is predicted after 10 days, while it advises selling when a decrease in prices is predicted after 10 days.

The trading recommendations produced by our model for a subset of equities utilized in both the training and testing phases are depicted in Figure 3. It is important to note that since these equities were part of the model's training data, the recommendations pertain to a time period not previously encountered by the model (specifically, the year 2023).

In Figure 4, we present two singular cases concerning two pharmaceutical sector stocks exhibiting opposite and presumably unpredictable behavior, which our model appears to have predicted. On 13 February 2023, Frequency Therapeutics Inc. shares plummeted by over 75% to reach an all-time low in early trading. This setback followed the regenerative medicine company's announcement of the termination of its primary program after its failure in a Phase 2b study. Notably, our model predicted this drop.

Conversely, on 13 March 2023, Provention Bio's shares soared following its acquisition agreement with French pharmaceutical company Sanofi for $2.9 billion, or $25 a share. The biopharmaceutical company, specializing in autoimmune diseases, experienced a 258%

surge in its stock to $24, up from Friday's closing price of $6.70. Our model also predicted this surge.

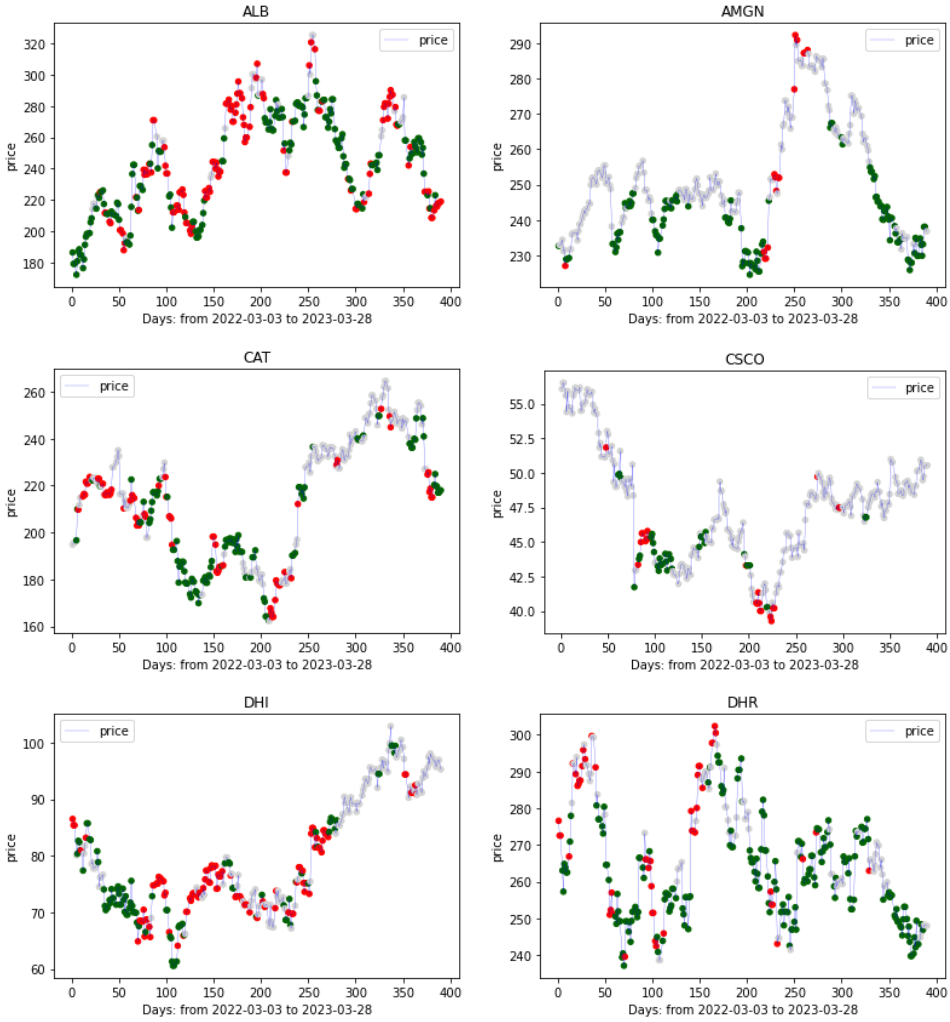

Figure 1. Trading recommendations for the period March 2022–March 2023 generated by the model for a subset of stocks (ALB, AMGN, CAT, CSCO, DHI, DHR). These stocks were not used during the training and testing of the model. As usual, red dots denote a forecasted drop in prices, while green dots denote a forecasted rise in prices.

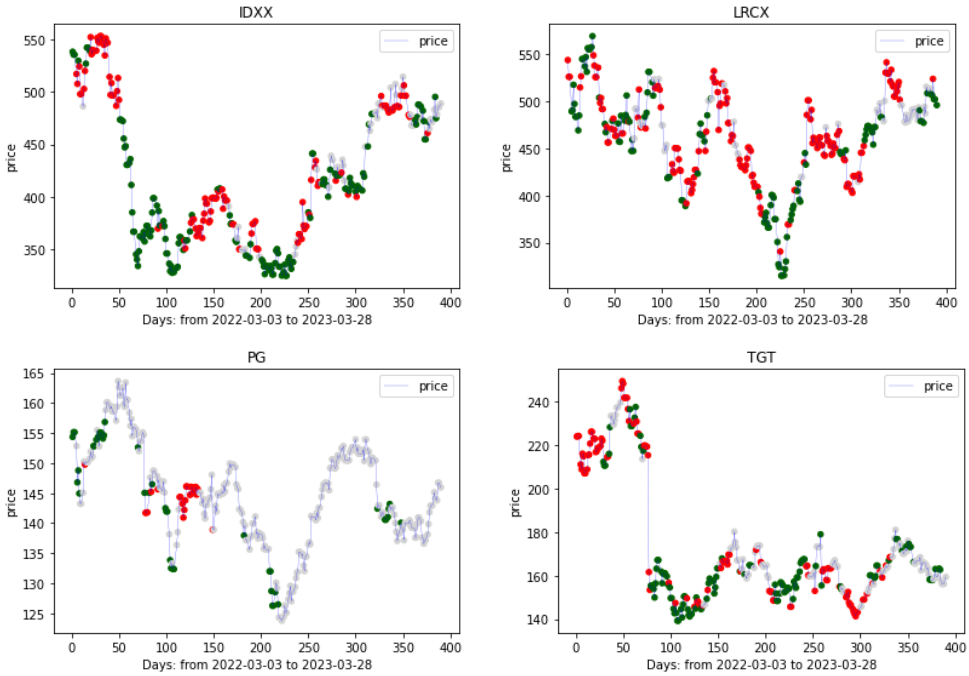

Figure 2. Trading recommendations for the period March 2022–March 2023 generated by the model for another subset of stocks (IDXX, LRCX, PG, TGT). These stocks were not used during the training and testing of the model. As usual, red dots denote a forecasted drop in prices, while green dots denote a forecasted rise in prices.

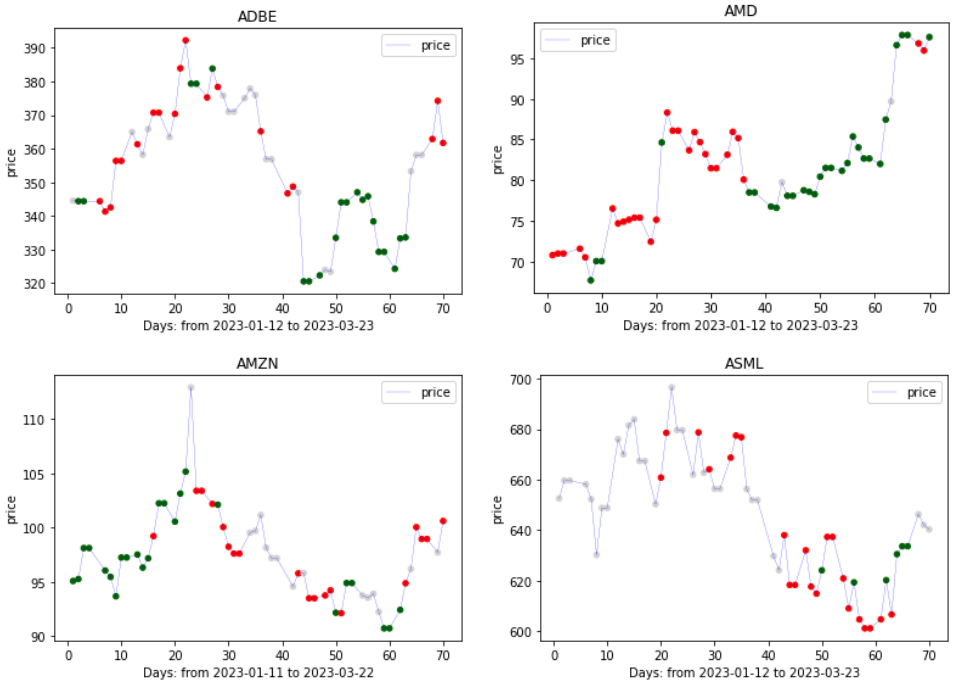

Figure 3. *Cont.*

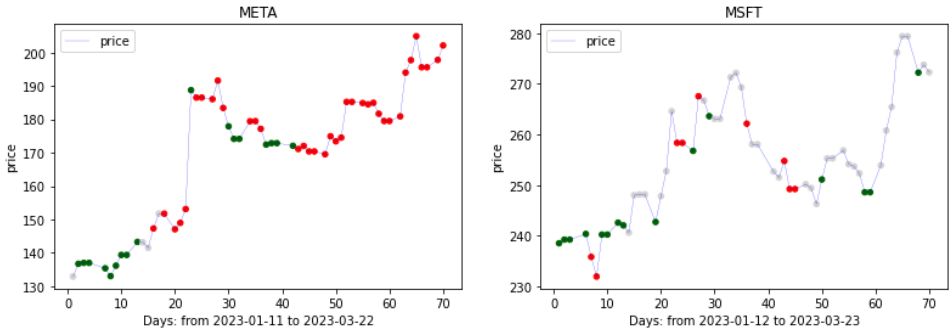

Figure 3. Trading recommendations for the period January to March 2023 generated by the model for a subset of stocks (ADBE, AMD, AMZN, ASML, META, MSFT) used during the training and testing of the model. The recommendations pertain to a time period not previously encountered by the model. As usual, red dots denote a forecasted drop in prices, while green dots denote a forecasted rise in prices.

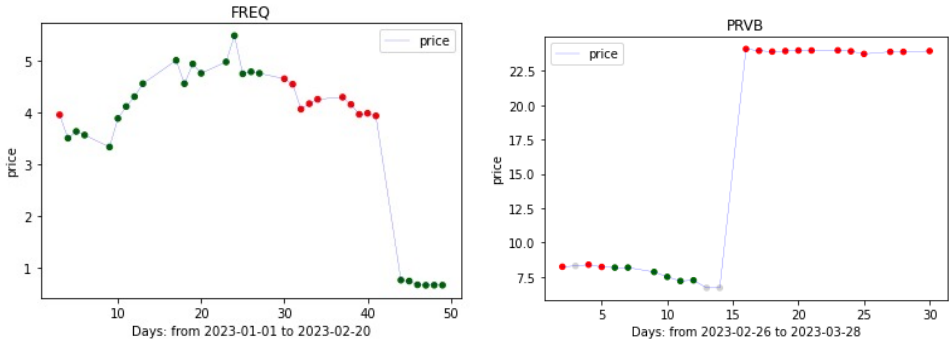

Figure 4. On the left: in early trading on 13 February 2023, the shares of Frequency Therapeutics Inc. experienced a decline of over 75% and reached an all-time low. The setback occurred after the regenerative medicine company announced the termination of its primary program, subsequent to its failure in a Phase 2b study. Interestingly enough, according to our model this drop was predictable. On the right: Provention Bio's shares surged on Mar 13th 2023 following its agreement to be acquired by French pharmaceutical company Sanofi for $2.9 billion, or $25 a share. The biopharmaceutical company, which specializes in autoimmune diseases, saw its stock rise by 258% to $24, up from Friday's closing price of $6.70. Again, interestingly enough, according to our model this surge was predictable.

7. Discussion

Predicting the direction of stock market prices is a challenging task that has been the subject of much research in the field of finance and economics. One of the most popular methods for making stock market predictions is the use of machine learning, specifically tree-based algorithm.

In this paper we aimed to utilize advanced machine learning models to predict significant fluctuations in asset prices in the stock market. We evaluated various technical indicators to train a set of classifier models. The performance of the models were evaluated using several metrics. The results showed that our best model is an Extra Trees Classifier that achieved an accuracy of 86.1%, indicating that the model outperforms the more classical Random Forest model and could predict significant fluctuations in asset prices.

An Extra Trees Classifier is an ensemble machine learning method that consists of a collection of decision trees. Each tree is trained on a random subset of the data, and during prediction, the Extra Trees Classifier aggregates the predictions of each individual tree to arrive at the final output. One of the advantages of using an Extra Trees Classifier for stock market predictions is its ability to handle large data sets with a large number of input features. The Extra Trees algorithm was used to predict the direction of stock market prices

by training the model on preprocessed historical stock market data. The input features for the model include historical stock closing prices, trading volumes, and various technical indicators. The results showed that the most important features were BOLL L with an importance score of 0.0748, followed closely by BOLL M, ADX, BOLL U, FWMA, TSI, and MACD. The least important features are SLOPE AD and SLOPE PFD with importance scores of 0.0536 and 0.0574, respectively. The results suggest that BOLL L and BOLL M may be the most informative features for predicting the price direction. The output of the model is a prediction of whether the stock market will significantly rise or fall in a 10 trading days window.

While Extra Trees Classifiers can be useful for making stock market predictions, it is important to acknowledge that they are not always entirely accurate. The stock market is a complex and dynamic system that is influenced by various factors, some of which are difficult to predict. Furthermore, stock market predictions are inherently subject to volatility and uncertainty, and should be considered as a tool for guidance rather than a definitive answer. In summary, the Extra Trees Classifier algorithm can be a powerful tool for predicting stock market directions by training on historical stock market data and using various input features such as technical indicators and historical stock prices. However, it is crucial to recognize the limitations of these predictions and use them as a guide.

8. Conclusions

This work introduced an Extra Trees Classifier model optimized for short-term stock market return forecasting. The results demonstrated its effectiveness in achieving a high accuracy of 86.1%, outperforming classical methods such as Random Forests. This highlights the promise of classification-based machine learning techniques for stock prediction as an improvement over prevailing regression approaches. However, while these models can provide useful guidance, their probabilistic forecasts have limitations. The intricate dynamics of financial markets imply inherent uncertainty in price predictions. Therefore, model outputs should be considered predictive rather than definitive. This study contributes an initial exploration into classification algorithms for stock forecasting, opening up avenues for further research into hybrid models and ensemble techniques to enhance market insights.

Funding: This research received no external funding.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: For the computation of the technical indicators of Section 2, we used Pandas TA—A Technical Analysis Library in Python 3 [15]. The author acknowledges supercomputing resources and support from ICSC—Centro Nazionale di Ricerca in High Performance Computing, Big Data and Quantum Computing – and hosting entity, funded by European Union—NextGenerationEU. The author wishes to thank Pierluca Sangiorgi for suggestions, discussions and for testing the model *on the field*, the friends from *Shhtonks* and *B & Mercati* groups for enjoying together bull and bear markets and Matteo "il Don" for daily suggestions.

Conflicts of Interest: During the preparation of this work the author used ChatGPT OpenAI and Claude 2 Anthropic in order to improve readability and language. After using this tool, the author reviewed and edited the content as needed and takes full responsibility for the content of the publication.

Appendix A. List of Stocks Used for Training the Model

AAPL, ABT, ADBE, AKTX, AMD, AMTI, AMZN, ANET, APAM, APPN, ASML, ATHX, ATOM, ATVI, AVLR, AXTI, BA, BKNG, BRK-B, BTWNW, BYND, CCI, CDNS, CLVS, COUP, CRWD, CTXR, DBX, DIS, DOCU, EA, EOLS, EQX, ETSY, FB/META, FICO, FLGT, FREQ, FSLY, FVRR, GOOGL, GSV, HMI, HUIZ, ICD, ILMN, INTC, INTU, ISRG, IZEA, JD, JMIA, JMP, JNJ, KEYS, KO, KOPN, LODE, LTRN, LYL, MA, MELI, MGNI, MSFT, MTSL, NBIX, NCTY, NET, NFLX, NVDA, NVTA, OBSV, OKTA, OLED, OPEN, PACB, PHVS, PINS,

PLTR, PODD, PYPL, QCOM, RENN, ROKU, SABR, SE, SEDG, SHOP, SINO, SNAP, SNE, SNOW, SPCE, SQ, SQM, TAOP, TDOC, TMO, TRIP, TSLA, TSM, TTD, TWLO, TWST, U, UAMY, UBX, UPST, UUUU, V, VTGN, WATT, WCC, WMT, WRAP, XNET, YNDX, ZBRA, ZM, ZNGA.

References

1. Malkiel, B.G.; Fama, E.F. Efficient capital markets: A review of theory and empirical work. *J. Financ.* **1970**, *25*, 383–417. [CrossRef]
2. Jensen, M.C. Some anomalous evidence regarding market efficiency. *J. Financ. Econ.* **1978**, *6*, 95–101. [CrossRef]
3. Avery, C.N.; Chevalier, J.A.; Zeckhauser, R.J. The CAPS prediction system and stock market returns. *Rev. Financ.* **2016**, *20*, 1363–1381. [CrossRef]
4. Christoffersen, P.F.; Diebold, F.X. Financial asset returns, direction-of-change forecasting, and volatility dynamics. *Manag. Sci.* **2006**, *52*, 1273–1287. [CrossRef]
5. Hellstrom, T.; Holmstromm, K. Predictable Patterns in Stock Returns; Technical Report Series IMa-TOM, 1997-09; 1998. Available online: https://api.semanticscholar.org/CorpusID:150923793 (accessed on 3 November 2023)
6. Saha, S.; Routh, S.; Goswami, B. Modeling Vanilla Option prices: A simulation study by an implicit method. *J. Adv. Math.* **2014**, *6*, 834–848.
7. Widom, J. Research problems in data warehousing. In Proceedings of the Fourth International Conference on Information and Knowledge Management, CIKM '95, Baltimore, MD, USA, 29 November–2 December 1995; ACM: New York, NY, USA, 1995; pp. 25–30.
8. Kumbure, M.M.; Lohrmann, C.; Luukka, P.; Porras, J. Machine learning techniques and data for stock market forecasting: A literature review. *Expert Syst. Appl.* **2022**, *197*, 116659. [CrossRef]
9. Kara, Y.; Boyacioglu, M.A.; Baykan, C. Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul stock exchange. *Expert Syst. Appl.* **2011**, *38*, 5311–5319. [CrossRef]
10. Adebiyi, A.A.; Adewumi, A.O.; Ayo, C. Comparison of ARIMA and artificial neural networks models for stock price prediction. *J. Appl. Math.* **2014**, *2014*, 614342. [CrossRef]
11. de Faria, E.L.; Albuquerque, M.P.; Gonzalez, J.L.; Cavalcante, J. Predicting the Brazilian stock market through neural networks and adaptive exponential smoothing methods. *Expert Syst. Appl.* **2009**, *36*, 12506–12509. [CrossRef]
12. Basak, S.; Kar, S.; Saha, S.; Khaidem, L.; Dey, S.R. Predicting the direction of stock market prices using tree-based classifiers. *N. Am. J. Econ. Financ.* **2019**, *47*, 552–567. [CrossRef]
13. Bruno, A.; Pagliaro, A.; La Parola, V. Application of Machine and Deep Learning Methods to the Analysis of IACTs Data. In *Intelligent Astrophysics*; Zelinka, I., Brescia, M., Baron, D., Eds.; Emergence, Complexity and Computation, Volume 39; Springer: Berlin/Heidelberg, Germany, 2021; pp. 115–136.
14. Pagliaro, A.; Cusumano, G.; La Barbera, A.; La Parola, V.; Lombardi, S. Application of Machine Learning Ensemble Methods to ASTRI Mini-Array Cherenkov Event Reconstruction. *Appl. Sci.* **2023**, *13*, 8172. [CrossRef]
15. Twopirllc. Pandas-TA: Technical Analysis Indicators for Pandas. Available online: https://twopirllc.github.io/pandas-ta/ (accessed on 3 November 2023).
16. Appel, G. The MACD Momentum Indicator. *Tech. Anal. Stock. Commod.* **1985**, *3*, 84–88.
17. ProRealCode. Schaff Trend Cycle (STC). Available online: https://www.prorealcode.com/prorealtime-indicators/schaff-trend-cycle2/ (accessed on 3 November 2023).
18. Williams, L. *How I Made One Million Dollars Last Year Trading Commodities*; FutureBooks: Singapore, 1973.
19. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
20. Geurts, P.; Ernst, D.; Wehenkel, L. Extremely randomized trees. *Mach. Learn.* **2006**, *63*, 3–42. [CrossRef]