High Accuracy Stock Trend Prediction using XGBoost Model

Alex Chen
University of Montreal Master of Science in Computer Science
alex.chen.1@umontreal.ca

Abstract

This paper presents an investigation into stock price prediction using an XGBoost model trained on SPDR S&P 500 ETF trust (SPY) data from the US market retrieved from Yahoo Finance. The model is iteratively trained and validated, focusing on predicting the low, high and close prices of the current day based on the open price and historical daily stock data.

One interesting property discovered was the model's stunning accuracy in forecasting if the low, high or close prices of today are higher or lower than yesterday's when the market opens. For most SPY500 stocks, they reach more than 90% of accuracy if the change predicted is above 3%. This property is significant as it could offer real-world financial gains. The research also explores a few techniques to improve the model's performance. Notably, using a moderate amount of recent data achieves comparable results than using the entire historical dataset. Different training data combinations are assessed, with ["open", "high", "low", "close"] emerging as the most effective for predicting closing prices. By addressing considerations such as data selection and model parameter tuning, the study offers insights into optimizing accuracy and reliability in stock price prediction using XGBoost.

1 Introduction

Stock price prediction has long been a challenging and high-stakes endeavor for investors, analysts, and financial institutions. The dynamic and often unpredictable nature of financial markets makes accurate forecasting a complex task. However, the integration of machine learning techniques has brought a new dimension to this domain, offering the potential to enhance predictive accuracy and provide valuable insights. Machine learning leverages historical stock price data, market indicators, and other relevant features to develop predictive models. These models aim to capture intricate patterns and trends that might be difficult for human analysts to discern. The advantage of machine learning lies in its ability to process vast amounts of data and adapt to changing market conditions in real time.

One common approach to stock prediction involves the use of time series analysis, where historical price data is analyzed to identify recurring patterns and trends. Algorithms such as autoregressive integrated moving average, ARIMA (Ariyo et al., 2014), recurrent neural networks, RNNs (Selvin et al., 2017), and more recently, transformers (Liu et al., 2019), have demonstrated effectiveness in modeling the sequential nature of stock prices. Decision trees are a robust tool for stock prediction when solely relying on stock data. They can analyze historical stock prices and identify patterns in these data points by narrowing the decision tree structure that leads to the best accuracy. However, challenges abound. Financial markets are influenced by a multitude of factors, including global events, economic indicators, and geopolitical shifts, making predictions susceptible to sudden disruptions. Moreover, overfitting, data quality, and the efficient market hypothesis pose additional hurdles for accurate predictions.

One of the prominent models in stock prediction is XGBoost (Chen and Guestrin, 2016). This technique, an extension of gradient boosting, has gained substantial traction in the realm of stock prediction due to its ability to handle intricate relationships within data, providing a fresh perspective on market analysis. This paper investigates stock market prediction using an XGBoost model trained on the SPDR S&P 500 ETF trust (SPY) historical daily stock data from the US market. It evaluates the influence of training data volume, data combinations and hyperparameters on model's accuracy. An interesting property discovered about the model was its ability in forecasting if the low, high or close prices of today are higher or lower than yesterday's (trend prediction) with very high accuracy. It has promising potential to offer real-world financial gains to investors.

2 Background

2.1 Additive Training

XGBoost stands for "Extreme Gradient Boosting". It uses tree ensembles coupled with additive training. The model is in the form of Equation 1.

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F} \tag{1}$$

where K is the number of trees, f_k is a function in the functional space $\mathcal{F}$, and $\mathcal{F}$ is the set of all possible trees. The objective function to be optimized is given by Equation 2.

$$\text{obj}(\theta) = \sum_{i}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \omega(f_k) \tag{2}$$

where $\omega(f_k)$ is the complexity of the tree, and f_k represents the structure and the leaf scores of each tree.

Learning the tree structure is much harder than traditional optimization problems. It is impossible to learn all the trees at once. Instead, an additive strategy is used where the trees that are already learnt are fix, then one new tree is added at a time. The prediction value at step t as $\hat{y}_i^{(t)}$ can be written in Equation 3.

$$\begin{aligned}
\hat{y}_i^{(0)} &= 0 \\
\hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \\
\hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\
&\dots \\
\hat{y}_i^{(t)} &= \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned} \tag{3}$$

If we consider using mean squared error (MSE) as our loss function, the objective becomes Equation 4.

$$\begin{aligned}
\text{obj}^{(t)} &= \sum_{i=1}^{n} (y_i - (\hat{y}_i^{(t-1)} + f_t(x_i)))^2 + \sum_{i=1}^{t} \omega(f_i) \\
&= \sum_{i=1}^{n} [2(\hat{y}_i^{(t-1)} - y_i) f_t(x_i) + f_t(x_i)^2] \\
&+ \omega(f_t) + \text{constant}
\end{aligned} \tag{4}$$

If the second order Tylor expansion was taken to the loss function and all the constants were removed, the specific objective at step t becomes Equation 5.

$$\sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \omega(f_t) \tag{5}$$

Where g_i and h_i are defined in Equation 6.

$$\begin{aligned}
g_i &= \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)}) \\
h_i &= \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})
\end{aligned} \tag{6}$$

And the regularization term $w(f)$ in XGBoost is defined in Equation 7.

$$\omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \tag{7}$$

T is the number of leaves, γ and λ are regularization constants.

2.2 The Structure's Score

After re-formulating the tree model, the objective value with the t-th tree can be written in Equation 8.

$$\begin{aligned}
\text{obj}^{(t)} &\approx \sum_{i=1}^{n} [g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \\
&= \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \lambda) w_j^2] + \gamma T
\end{aligned} \tag{8}$$

where $I_j = \{i | q(x_i) = j\}$ is the set of indices of data points assigned to the j-th leaf. Notice that the index of the summation in the second line was changed, because all the data points on the same leaf get the same score. The expression could be further compressed into Equation 9 by defining $G_j = \sum_{i \in I_j} g_i$ and $H_j = \sum_{i \in I_j} h_i$.

$$\text{obj}^{(t)} = \sum_{j=1}^{T} [G_j w_j + \frac{1}{2} (H_j + \lambda) w_j^2] + \gamma T \tag{9}$$

In Equation 9, considering that w_j's are independent of each other, and the form $G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2$ is quadratic. The best w_j of a given structure $q(x)$ and the best objective objective can be summarized in Equation 10.

$$\begin{aligned}
w_j^* &= -\frac{G_j}{H_j + \lambda} \\
\text{obj}^* &= -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T
\end{aligned} \tag{10}$$

With Equation 10, it is possible to measure how good a tree structure $q(x)$ is.

2.3 Objective

The objective of this paper is to train an XG-Boost model that is able to predict the stock price of today ($\hat{y}$) with the highest accuracy possible. The model should be only trained to predict one of the close, high, or low price of today at a time based only on today's open price and all historical daily stock data. In another word, if the model has been given today's open price when the market opens, and all historical data, it should be able to predict the high, low or close prices of today accurately. These predictions, if precise enough, could offer valuable daily forecast that would help investors expand returns and avoid risks.

3 Methods

In this paper, the data used for training, validation and prediction are all US market data. In the previous research conducted by Shi et al. (Zhuang-wei Shi and Wu, 2023), a similar XGBoost model was built that relied on the stock data from the Chinese market. The stock that this paper used is the SPDR S&P 500 ETF trust, with the stock symbol of SPY. It is an investment fund designed to track the performance of the S&P 500 Index, which is a widely-followed benchmark that measures the performance of 500 large-cap U.S. companies. It serves as an indicator of the overall health and performance of the U.S. stock market. SPY provides investors with exposure to a diversified portfolio of large-cap U.S. stocks and is one of the most popular and widely traded exchange-traded funds.

The performance of stock market prediction AI can be significantly influenced by unpredictable events, such as natural disasters, political unrest, or economic crises. They can introduce volatility and uncertainty into the market, making it challenging for AI algorithms to accurately forecast stock prices. Unpredictable events can disrupt traditional market patterns and render historical data less reliable. Moreover, these events may introduce new factors that were not previously considered by the AI system, further affecting its performance. Using an exchange-traded fund like SPY instead of individual stocks may mitigate the risks of unpredictable events. SPY provides diversification across different sectors, reducing the impact of any single event on the overall portfolio. This can help spread risk and potentially provide more stability in uncertain market conditions.

The main programming language for this project is Python 3. Python is a popular and versatile programming language that offers numerous advantages for machine learning projects. Firstly, its simplicity and readability make it easy for both beginners and experts to write and understand complex machine-learning projects. Python also boasts a vast array of machine learning libraries and frameworks, such as TensorFlow, scikit-learn, PyTorch, etc., which simplify the implementation of complex machine learning algorithms and data manipulation. The paper explores the XGBoost model (xgboost developers, 2022), which is free and open-sourced. Pandas was used for data manipulation, and MatPlotLib was used for plotting.

The machine used to train this AI is MacBook 2021 with M1 chip and 16GB of memory.

3.1 Preprocessing and Data Retrieving

Yahoo Finance was used to retrieve the historical data of SPY. It is free and provides complete historical data up to 1990s since SPY first IPOed. More than 7000 daily datasets have been retrieved from Yahoo Finance. They cover SPY's daily market data from March 1993 up to the time when this paper was written, July 2023. Each dataset after preprocessing consists of the trading information of one day, and it is in the format of Table 1.

Table 1: CSV Data Format

trade_date	open	high	low	close
19930201	43.96875	44.25	43.96875	44.25

The "trade_date" is the date in YYYYMMDD format under which the trading information is found. "open" is the stock price when the market opens. "high" is the highest trading price the stock has reached during that day, "low" is the lowest trading price the stock has reached during that day, "close" is the stock price at the end of the day, and "volume" is the stock units traded during that day.

The raw data retrieved from Yahoo Finance was then further processed to be fed into XGBoost. They are converted to a Pandas data frame then to a python series in the format of Table 2.

Table 2: Series format for XGBoost

	var1(t-2)	var2(t-2)	var1(t-1)	var2(t-1)	var1(t)	var2(t)
2	2022-01-08 Open	2022-01-08 Training Price	2022-01-09 Open	2022-01-09 Training Price	2022-01-10 Open	2022-01-10 Training Price
3	2022-01-09 Open	2022-01-09 Training Price	2022-01-10 Open	2022-01-10 Training Price	2022-01-11 Open	2022-01-11 Training Price
4	2022-01-10 Open	2022-01-10 Training Price	2022-01-11 Open	2022-01-11 Training Price	2022-01-12 Open	2022-01-12 Training Price
5	2022-01-11 Open	2022-01-11 Training Price	2022-01-12 Open	2022-01-12 Training Price	2022-01-13 Open	2022-01-13 Training Price

Table 2 provides 4 rows of dataset as an example. It displays the final data format for the XGBoost training. $var1(t)$ and $var2(t)$ is the open and training price of the current day respectively. $var1(t-1)$ and $var2(t-1)$ is the open and training price of yesterday respectively. $var1(t-2)$ and $var2(t-2)$ is the open and training price of the day before yesterday respectively. The data from the same day are colored in the same shade of grey, and they are arranged in a staircase shape.

Once the data have been processed to the XGBoost format, they were split into train and test sets. The test set contains the latest 400 datasets, and the train set is the rest of the data.

3.2 Training and Validation

The training and validation process is as the following:

```
y_hats = []
For 1 to len(test_set):
    x_test = test_set[i][var1(t-2),
        var2(t-2), var1(t-1),
        var2(t-1), var1(t)]
    x_train =
        training_set[var1(t-2),
        var2(t-2), var1(t-1),
        var2(t-1), var1(t)]
    y_train = training_set[var2(t)]
    model =
        xgb.XGBRegressor(model_parameters)
    model.fit(x_train, y_train)
    y_hat = model.predict(x_test)
    y_hats.append(y_hat)
    training_set.append(test_set[i])
```

For each iteration, the workflow is shown in Figure 1 below.

In the training process, the open price is $var1$, and $var2$ is the training price that the model is trying to predict (low, high or close). The process is taking a step-wise approach, where the historical data of all previous days until the test date were used as the training data, and then the training price of the test date was predicted. 400 iterations have been executed.

For example, if the model were to predict the high price, and today's date is August 1, 2023. The first testing iteration would be 400 trading days ago, which is Jan 14, 2022. For that iteration, all the open and high prices from Jan 08, 1999 up to Jan 13, 2022 would be taken to construct the training table shown in Figure 1 and fed into the XGBoost model for training. In total, there are more than 7000 training data. Then, the open and high prices of Jan 13, 2022 and Jan 12, 2022, and the open price of Jan 14, 2022 are used to construct the test table in Figure 1. The test table is then fed into the trained model to predict the high price of Jan 14 ($\hat{y}$). The predicted high price is compared with the actual high price (y). This step-wise iteration is repeated for 400 times to predict the high prices from Jan 14, 2022 all the way up to August 1, 2023. 400 $\hat{y}$'s have been produced, then they are compared to the actual y data from the test set to see the model performance.

4 Experiments

In order to evaluate the model's performance, the following evaluation metrics were used: mean absolute error (MAE), mean square error (MSE), root of mean square error (RMSE), mean absolute percentage error (MAPE) and R2.

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|\hat{y}_t - y_t| \tag{11}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(\hat{y}_t - y_t)^2} \tag{12}$$

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}|\frac{\hat{y}_t - y_t}{y_t}| \tag{13}$$

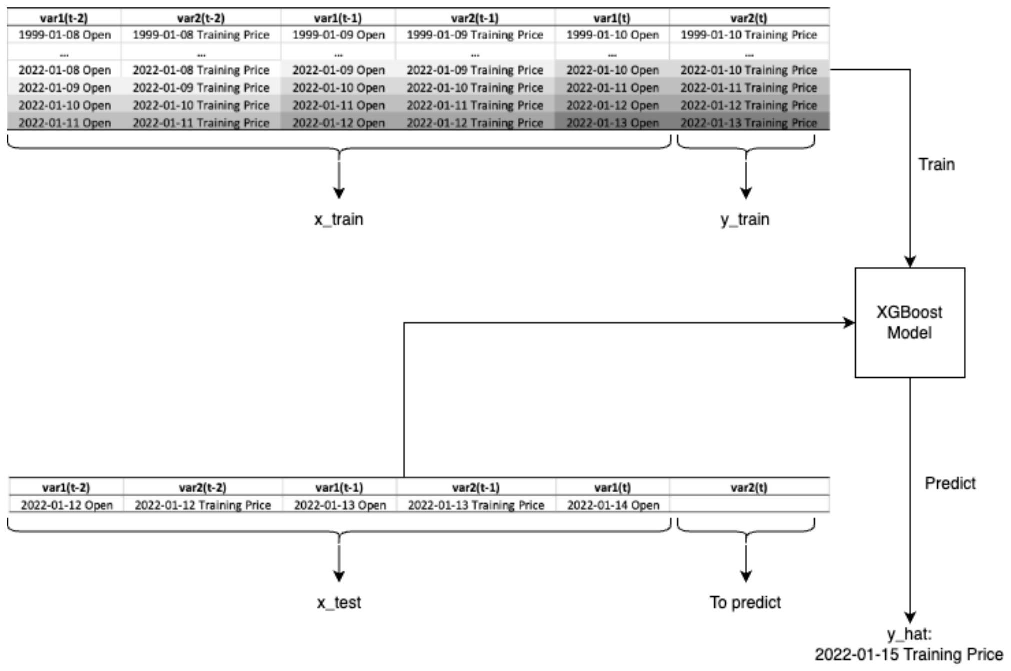
$$R^2 = \frac{\sum_{t=1}^{n}||\hat{y}_t - \bar{y}_t||^2}{\sum_{t=1}^{n}||y_t - \bar{y}_t||^2} \tag{14}$$

4.1 Fine Tuning

The XGBoost model was trained and iterated 400 times with the latest stock data and the default model parameters. This run is referred to as the "baseline". The baseline performance is shown in Table 3.

Table 3: Baseline Model Performance

Training Price	MSE	RMSE	MAE	R2	MAPE
low	9.62505	3.10243	2.32689	0.884887	0.599633
high	7.04117	2.65352	2.01058	0.912691	0.508567
close	17.9325	4.23468	3.15842	0.797262	0.803541



var1(t-2)	var2(t-2)	var1(t-1)	var2(t-1)	var1(t)	var2(t)
1999-01-08 Open	1999-01-08 Training Price	1999-01-09 Open	1999-01-09 Training Price	1999-01-10 Open	1999-01-10 Training Price
...	...	...	...	...	...
2022-01-08 Open	2022-01-08 Training Price	2022-01-09 Open	2022-01-09 Training Price	2022-01-10 Open	2022-01-10 Training Price
2022-01-09 Open	2022-01-09 Training Price	2022-01-10 Open	2022-01-10 Training Price	2022-01-11 Open	2022-01-11 Training Price
2022-01-10 Open	2022-01-10 Training Price	2022-01-11 Open	2022-01-11 Training Price	2022-01-12 Open	2022-01-12 Training Price
2022-01-11 Open	2022-01-11 Training Price	2022-01-12 Open	2022-01-12 Training Price	2022-01-13 Open	2022-01-13 Training Price

var1(t-2)	var2(t-2)	var1(t-1)	var2(t-1)	var1(t)	var2(t)
2022-01-12 Open	2022-01-12 Training Price	2022-01-13 Open	2022-01-13 Training Price	2022-01-14 Open	

Figure 1: XGBoost One Iteration

Then, a grid search was performed with a range of model parameters to find the best set of parameters that would yield to an optimal performance. The search was done using SKlearn's GridSearchCV (scikit-learn developers (BSD License), 2023). The range of parameters used is listed in Table 4.

Table 4: Grid Search Parameter Values

Parameter Name	Parameter Values
objective	reg: squarederror
learning_rate	0.07, 0.08, 0.09
max_depth	7, 8, 9
min_child_weight	1, 2
subsample	0.5, 0.7, 0.9
colsample_bytree	0.5, 0.7, 0.9
n_estimators	140, 160, 180

Due to the computing limitation, only a small set of parameters were grid searched. In order to explore a further range of parameters, additional computing capacities need to be added. From this grid search, the optimal values were found out to be the ones listed in Table 5.

Table 5: Optimal Grid Search Parameter Values

Parameter Name	Optimal Parameter Value
objective	reg: squarederror
learning_rate	0.07
max_depth	7
min_child_weight	1
subsample	0.9
colsample_bytree	0.9
n_estimators	140

It is also observed that the model trains the fastest at 3 multiprocessing jobs. By using the parameters from Table 4, the model yields a better performance than the baseline as shown in Table 6. These parameters were used to perform the rest of the research.

Table 6: Model Performance with Optimal Parameters from the Grid Search

	MSE	RMSE	MAE	R2	MAPE
low	6.34874	2.51967	1.9549	0.924071	0.50463
high	7.85826	2.80326	2.08972	0.902559	0.531027
close	17.481	4.18103	3.06891	0.802366	0.780424

4.2 Number of Training Data Analysis

Another concern that was analyzed in this paper was if weather more training data would improve model's performance. In another word, does training with older historical stock data improves model's capabilities in predicting today's price? In order to answer that question, the training data were truncated from oldest to newest in a chunk of

200, and each performance is compared in Figure 2.

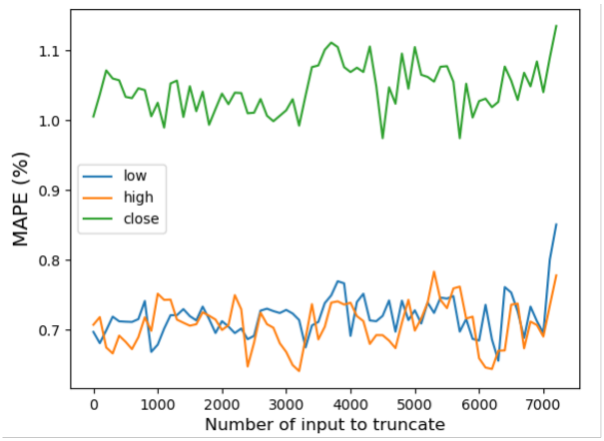


Figure 2: Model's Performance in Function of the Number of Inputs to Truncate

The X axis in Figure 2 represents the number of datasets excluded from training from earliest to latest. SPY since its IPO has about 7800 historical data. It can be observed that using all 7800 (0 on the x axis in Figure 3) data as training versus using only the 1000 latest data (6000 on the x axis in Figure 3) have similar performance. This is expected, since the data that's closer to the present should have more impacts. Although, using less than 400 of training data does increase the error rate significantly as can be seen in the sharp increase in MAPE after 7000 of input truncation. As a further recommendation, only 600 – 1000 training datasets are needed in order to avoid unnecessary overhead and increase the training speed.

4.3 Using Other Stock Data

The data retrieved from Yahoo Finance have other data than the ones listed in Table 1, such as:

- Volume: the number of stock units traded daily.
- Amount: the total number of stock units available for trading.
- Pre-close: the last trading price of the stock before the official closing of the trading session.
- Change: difference in its closing price between today and yesterday.
- Change percentage: difference in percentage in the closing price between today and yesterday.
- VWAP: The Volume-Weighted Average Price (VWAP) of a stock is the average price at which a specific volume of shares has been traded throughout the trading day, taking into account the trading volume at different price levels.
- Turnover ratio: proportion of shares traded in a given period relative to the total number of shares outstanding, indicating the stock's liquidity and market activity.

These pieces of additional information have been included in the training data in different combinations. Table 7 presents several data combinations for training along with their performance. The prediction price is the daily close. The performances are ranked from the best to worst. We can see that the combination of ['open', 'high', 'low', 'close'] yields the best performance. It indicates that data such as volume, amount, etc. hinders the model performance in predicting the closing price of the next day. Also, ['open', 'high', 'low', 'close'] performed better than just ['open', 'close'], which indicates that adding daily low and high prices in training helps with predicting the daily close. Dummy data combinations such as ['open', 'high', 'low', 'close'] with open, high, low of the previous day and ['open', 'high', 'low', 'close'], with open, high, low of being 0 were added and poor performances were expected.

4.4 Trend Prediction Analysis

One interesting property discovered about the trained XGBoost model using the fine-tune parameters and optimal data combination is its accuracy in predicting if the price of interest of today will be increasing or decreasing than yesterday's. It is still lacking in predicting precisely the exact price, but it has promising accuracy when it comes to trend prediction. For example, it was discovered that if the high price predicted today is 103, and the real high price of yesterday is 100, there is a very high likelihood that the real high price of today is higher than 100 as well. Similarly, if the high price predicted today is 98, chances are the real high price of today is lower than 100. This trend prediction property was further studied with the 400 iterations executed on SPY, and the result is shown in Figure 3.

The X axis represents the difference threshold between today's prediction price and yesterday's true price. The Y axis is the percentage of correct

Table 7: Different Data Combination and Their Performances

Data Combination	MSE	RMSE	MAE	R2	MAPE
['open', 'high', 'low', 'close']	13.60559	3.68858	2.63483	0.90475	0.68127
['open', 'close', 'high', 'low', 'change', 'pct_change']	13.74546	3.70749	2.88273	0.90377	0.74054
['open', 'close', 'high', 'low', 'pre_close', 'change', 'pct_change','vwap']	15.23896	3.90371	2.91031	0.89331	0.74376
['open', 'close', 'high', 'low', 'pre_close', 'change', 'pct_change','vwap', 'turnover_ratio']	15.23896	3.90371	2.91031	0.89331	0.74376
['open', 'close', 'high', 'low', 'pre_close', 'change', 'pct_change']	17.01546	4.12498	3.24435	0.88087	0.83359
['open', 'high', 'low', 'close', 'vol', 'amount']	16.76831	4.09491	3.24855	0.8826	0.83898
['open', 'close']	24.5601	4.95581	3.83915	0.82805	0.99019
['open', 'close', 'vol', 'amount']	35.55372	5.96269	4.48315	0.75108	1.15537
['open', 'high', 'low', 'close'] with open, high, low of the previous day	42.20278	6.49637	5.23187	0.70453	1.35249
['open', 'high', 'low', 'close'], with open, high, low of being 0	99.18435	9.95913	7.55001	0.3056	1.9601

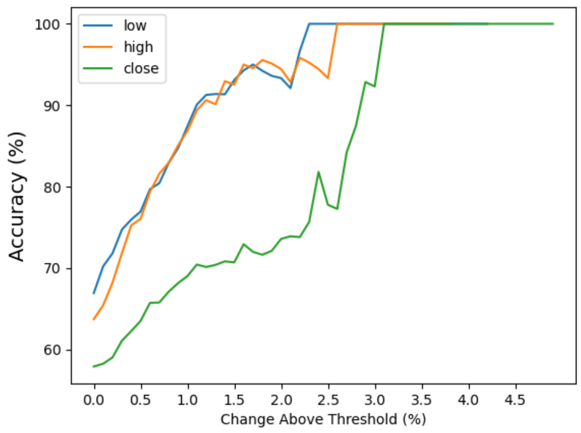


Figure 3: Model's Accuracy in Trend Prediction for SPY

predictions among 400 test sets. It can be observed in Figure 3 that the accuracy reaches 100% for low and high if the change predicted is more than 3%, whether increasing or decreasing. In another word, after the model has been trained, it is able to predict if the low, high or close price is going to be higher or lower than the ones of yesterday if today's open is provided. The higher the change predicted, the more likely it is correct. This offers very valuable insights into today's trend right when the market opens.

The same trend prediction accuracy analysis was made on all 500 SPY stocks. Figure 4 shows four of them.

As observed in Figures 3 and 4, the model approaches 100% accuracy when the change predicted is more than 3%. As seen in Figure 4, the accuracy varies in stocks. It would be interesting to know the stock that has the best trend prediction accuracy in order to maximize financial returns. In order to do that, the area under the curve (AUC) was calculated with the X axis fixed as in Figures 3 and 4 for all 500 stocks listed in SPY500. The model had to be trained independently on low, high and close prices, and on 500 stocks. In total, the model had to be trained 1500 times, which took more than 3 days. The 20 best ones are presented in Table 8 below by ranking the AUC of the daily high in a decreasing order, and their corresponding trend prediction accuracy graphs are presented in Appendix.

Table 8: 20 Most Accurate Stocks in Daily High Trend Prediction

Stock Symbol	Low MAPE	Low AUC	High MAPE	High AUC	Close MAPE	Close AUC
OGN	0.662	476.092	0.674	472.134	0.977	452.512
CAG	0.749	421.170	0.636	467.926	1.020	388.239
EIX	0.776	438.375	0.675	466.565	1.100	361.720
OMC	0.774	445.516	0.737	466.519	1.137	383.501
FTV	0.775	467.344	0.700	465.709	1.170	398.047
TEL	1.053	402.050	0.880	465.578	1.467	367.206
WMB	0.951	455.430	0.753	465.031	1.171	412.112
FRT	0.860	456.889	0.779	464.205	1.264	376.648
WRK	1.092	440.667	0.994	463.387	1.603	323.264
PPL	0.710	436.070	0.608	463.322	0.916	381.013
HAL	1.413	453.990	1.271	463.210	2.046	375.521
KO	0.683	431.033	0.641	462.624	0.881	364.134
JCI	0.932	454.752	0.855	462.478	1.274	378.209
EA	0.860	463.404	0.798	461.766	1.231	359.271
WY	0.981	452.304	0.841	461.670	1.397	348.584
ALLE	0.918	447.319	0.837	461.586	1.356	387.364
BXP	0.923	451.668	0.806	461.487	1.313	368.245
KDP	0.733	458.716	0.649	460.747	0.970	379.224
WAB	0.893	461.932	0.839	460.580	1.316	352.003

The XGBoost model trained in this paper still has a significant inaccuracy when predicting the price of interest. But it does offer valuable trend prediction especially when the change is large. These pieces of information could be used in making capital returns especially in day trading.

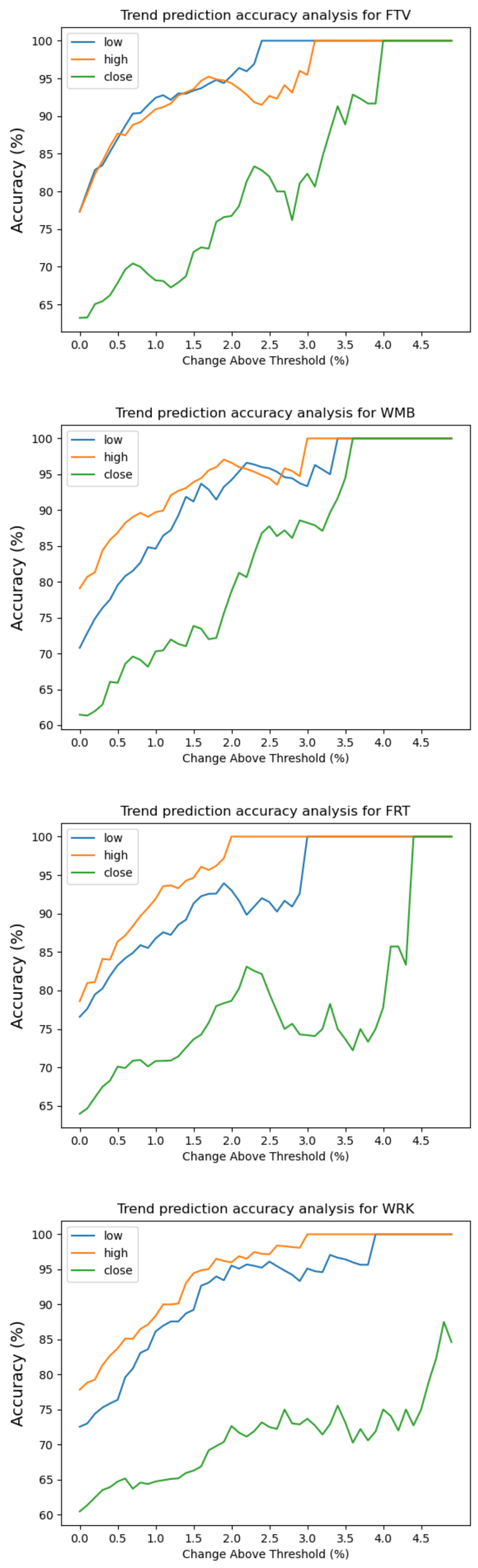


Figure 4: Model's Accuracy in Trend Prediction for Various Stocks

5 Conclusion

In conclusion, this study dives into the realm of stock price prediction through the lens of machine learning, employing an XGBoost model trained on SPDR S&P 500 ETF trust (SPY) data from the US market. The most important discovery made was the model has very high accuracy in forecasting if the low, high or close prices of today are higher or lower than yesterday's at the market open. The investigation into training data volume unveils a nuanced relationship between historical data and prediction performance. The discernment that a moderate volume of recent data yields comparable results to an extensive historical dataset underscores the model's adaptability to dynamic market conditions. Moreover, the exploration of diverse data combinations illuminates the pivotal role of pertinent features in refining predictive accuracy, with the ['open', 'high', 'low', 'close'] combination emerging as a robust choice.

LightGBM (Ke et al., 2017) is another gradient boosting framework that aims to improve training speed and reduce memory usage compared to XGBoost. It uses a histogram-based algorithm for efficient feature discretization and supports parallel and distributed training. From the experiments conducted by Microsoft, it seems that overall LightGBM has a better accuracy than XGBoost with significant memory reduction and speed increase. For further research, LightGBM could be used to perform a similar stock prediction experiment.

References

Adebiyi A Ariyo, Adewumi O Adewumi, and Charles K Ayo. 2014. Stock price prediction using the arima model. In 2014 UKSim-AMSS 16th international conference on computer modelling and simulation, pages 106–112. IEEE.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In 31st Conference on Neural Information Processing Systems (NIPS 2017). NIPS.

Jintao Liu, Hongfei Lin, Xikai Liu, Bo Xu, Yuqi Ren, Yufeng Diao, and Yang Liang. 2019. Transformer-based capsule network for stock movement prediction. In Proceedings of the first workshop on financial technology and natural language processing, pages 66–73.

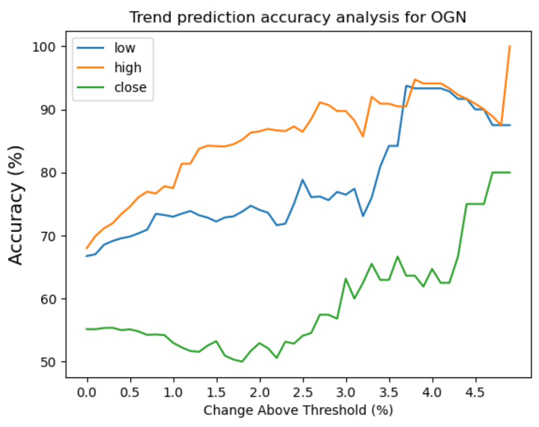
scikit-learn developers (BSD License). 2023. sklearn model selection GridSearchCV.

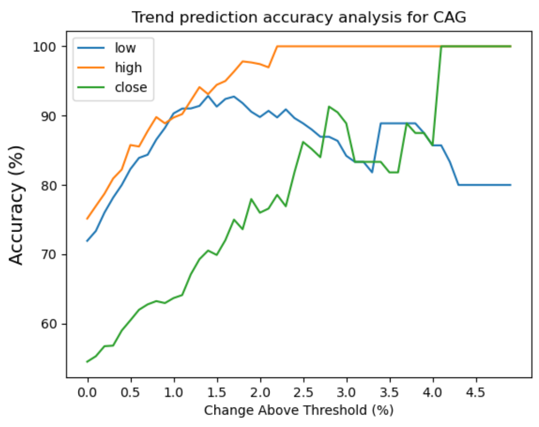
Sreelekshmy Selvin, R Vinayakumar, EA Gopalakrishnan, Vijay Krishna Menon, and KP Soman. 2017. Stock price prediction using lstm, rnn and cnn-sliding window model. In 2017 international conference on advances in computing, communications and informatics (icacci), pages 1643–1647. IEEE.

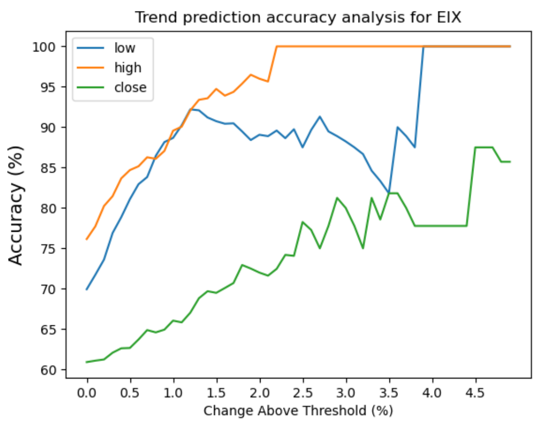
xgboost developers. 2022. Introduction to boosted trees.

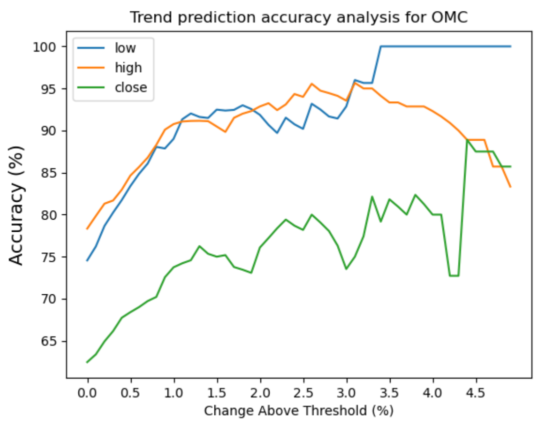
Guangliang Mo Zhuangwei Shi, Yang Hu and Jian Wu. 2023. Attention-based CNN-LSTM and XGBoost hybrid model for stock prediction. ArXiv.

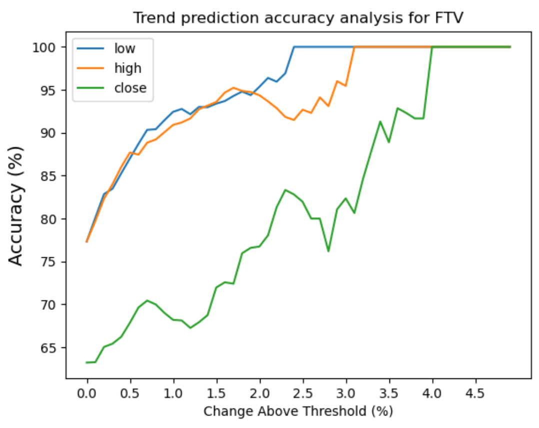
Appendix

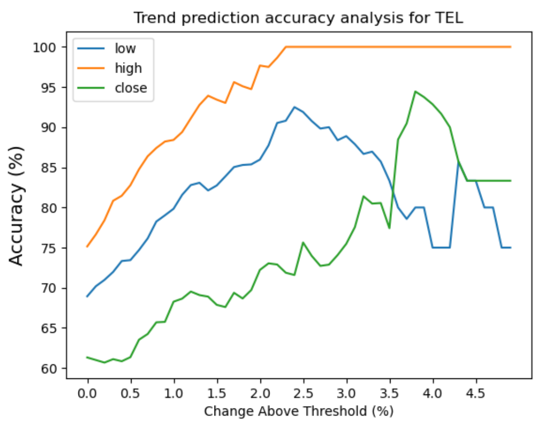

Trend prediction accuracy analysis for OGN
low
high
close
Accuracy (%)
Change Above Threshold (%)

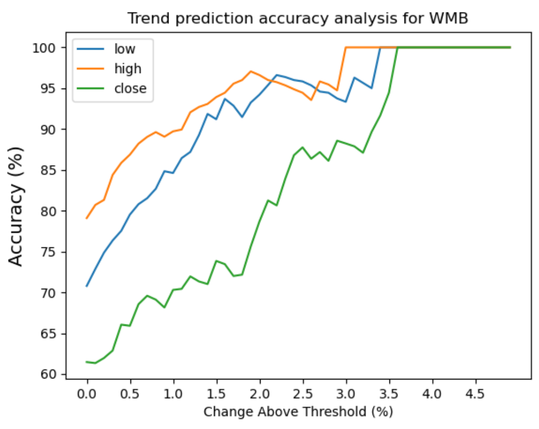

Trend prediction accuracy analysis for CAG
low
high
close
Accuracy (%)
Change Above Threshold (%)

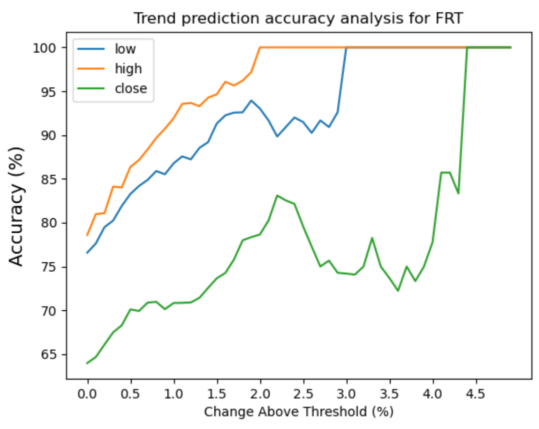

Trend prediction accuracy analysis for EIX
low
high
close
Accuracy (%)
Change Above Threshold (%)

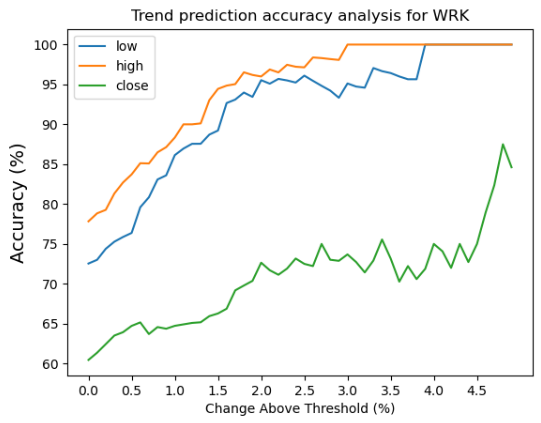

Trend prediction accuracy analysis for OMC
low
high
close
Accuracy (%)
Change Above Threshold (%)

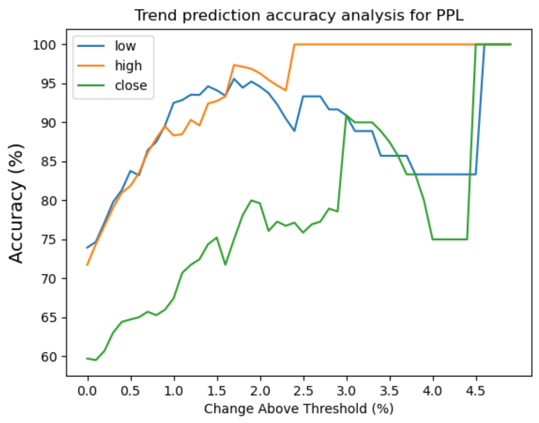

Trend prediction accuracy analysis for FTV
low
high
close
Accuracy (%)
Change Above Threshold (%)

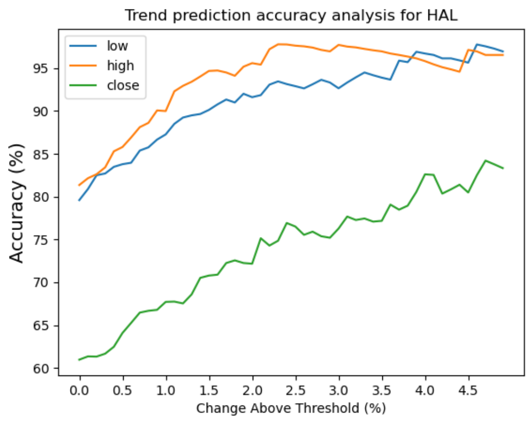

Trend prediction accuracy analysis for TEL
low
high
close
Accuracy (%)
Change Above Threshold (%)

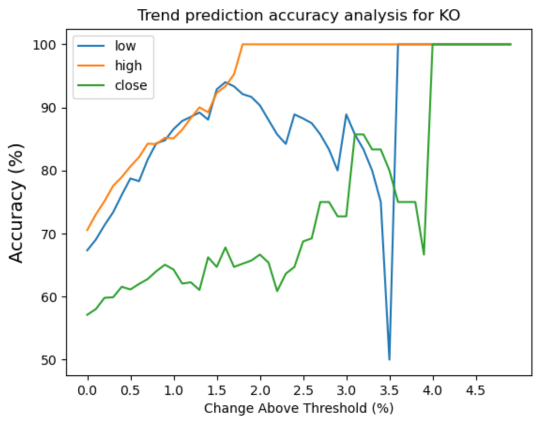

Trend prediction accuracy analysis for WMB
low
high
close
Accuracy (%)
Change Above Threshold (%)

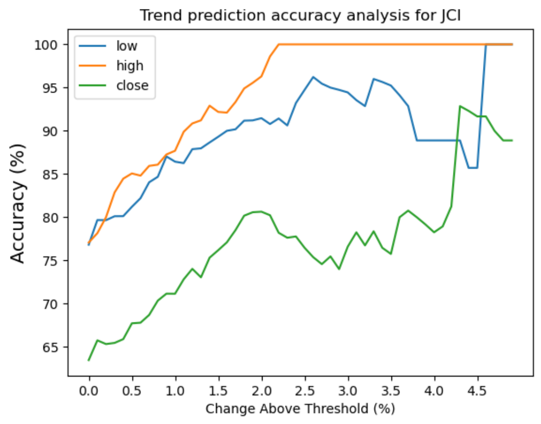

Trend prediction accuracy analysis for FRT
low
high
close
Accuracy (%)
Change Above Threshold (%)

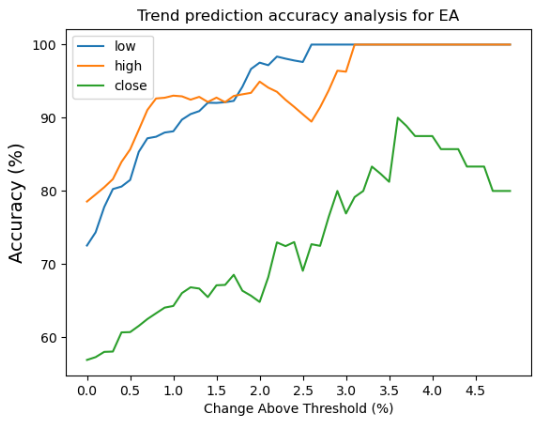

Trend prediction accuracy analysis for WRK
low
high
close
Accuracy (%)
Change Above Threshold (%)
60 65 70 75 80 85 90 95 100
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5

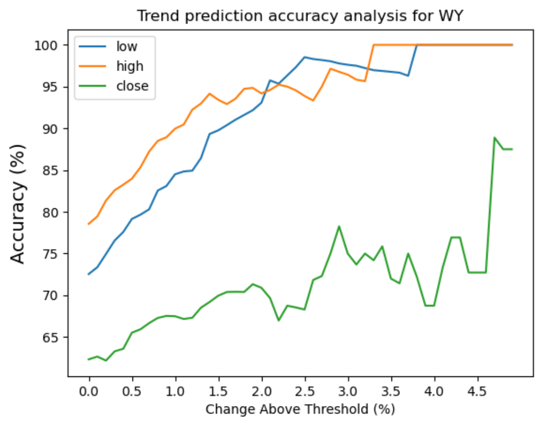

Trend prediction accuracy analysis for PPL
low
high
close
Accuracy (%)
Change Above Threshold (%)
60 65 70 75 80 85 90 95 100
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5

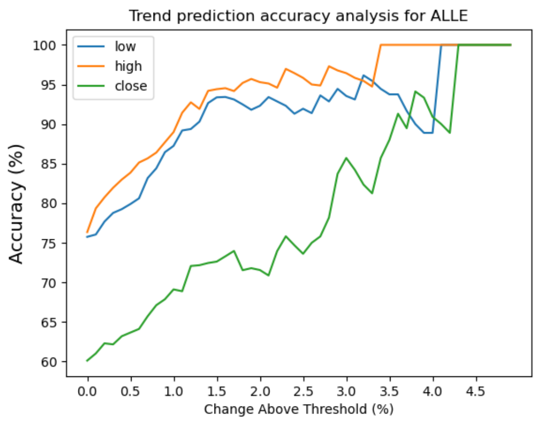

Trend prediction accuracy analysis for HAL
low
high
close
Accuracy (%)
Change Above Threshold (%)
60 65 70 75 80 85 90 95
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5

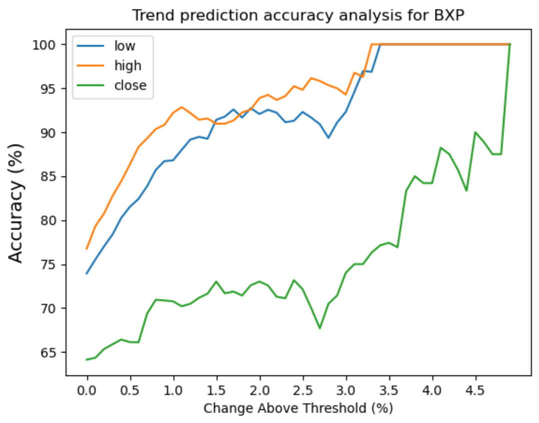

Trend prediction accuracy analysis for KO
low
high
close
Accuracy (%)
Change Above Threshold (%)
50 60 70 80 90 100
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5


Trend prediction accuracy analysis for JCI
low
high
close
Accuracy (%)
Change Above Threshold (%)
65 70 75 80 85 90 95 100
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5

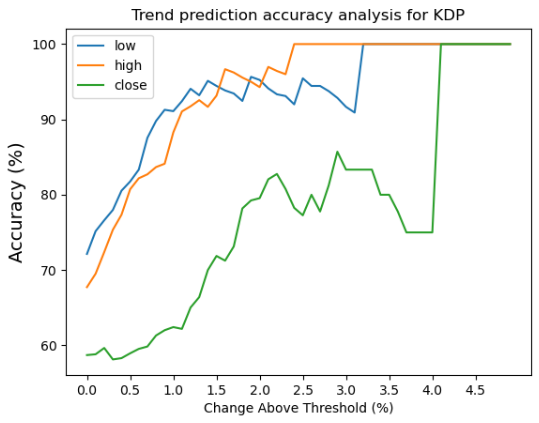

Trend prediction accuracy analysis for EA
low
high
close
Accuracy (%)
Change Above Threshold (%)
60 70 80 90 100
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5


Trend prediction accuracy analysis for WY
low
high
close
Accuracy (%)
Change Above Threshold (%)
65 70 75 80 85 90 95 100
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5

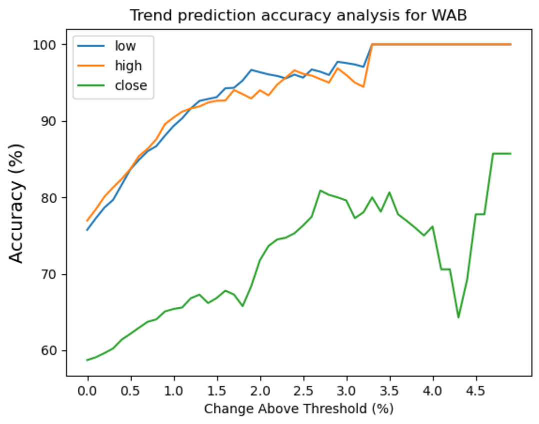

Trend prediction accuracy analysis for ALLE
low
high
close
Accuracy (%)
Change Above Threshold (%)
60 65 70 75 80 85 90 95 100
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5



Trend prediction accuracy analysis for BXP
low
high
close
Accuracy (%)
Change Above Threshold (%)



Trend prediction accuracy analysis for KDP
low
high
close
Accuracy (%)
Change Above Threshold (%)



Trend prediction accuracy analysis for WAB
low
high
close
Accuracy (%)
Change Above Threshold (%)