BOHR International Journal of Computer Science
2023, Vol. 2, No. 1, pp. 46–61
DOI: 10.54646/bijcs.2023.15
www.bohrpub.com

METHODS

Associating fundamental features with technical indicators for analyzing quarterly stock market trends using machine learning algorithms

Nicholas Moore and Sikha Bagui*

Department of Computer Science, University of West Florida, Pensacola, FL, United States

***Correspondence:**
Sikha Bagui,
bagui@uwf.edu

Received: 04 August 2023; **Accepted:** 19 August 2023; **Published:** 27 August 2023

The stock market is the primary entity driving every major economy across the globe, with each investment designed to capitalize on profit while decreasing its associated risks. As a result of the stock market's importance, there have been enumerable studies conducted with the goal of predicting the stock market through data analysis techniques including machine learning, neural networks, and time series analysis. This paper uses machine learning algorithms to perform stock market index classification using fundamental data while classifying the indices using technical indicators. The data were derived from Yahoo Finance on the top 100 indices in the NASDAQ stock market from January 2000 to December 2020.

Keywords: stock market, machine learning, technical indicators, fundamental analysis, NASDAQ

1. Introduction and related works

The health of every economy in the world, both major and growing, hinges on their market's stock prices, and predicting these stock prices is a growing area of interest for world governments, professional investors, and private citizens. Despite efforts to develop new techniques and strategies toward this goal, market volatility along with the non-linear high heteroscedasticity of market data present a model that is problematic to forecast (1). There are three main approaches to analyzing the stock market: technical, fundamental, and sentimental. Technical analysis attempts to determine future price change patterns using technical indicators, and these indicators include the opening price (open), daily highest price (high), daily lowest price (low), closing price (close), adjusted closing price (adjusted close), and the total volume (volume). Technical indicators are detailed in daily stock market reports and represent data efficiently for time series analysis (2).

Fundamental analysis uses the economic standing of a firm's yearly or quarterly reports to predict future stock value Nti et al., (3). Fundamental analysis is the focus of this paper. Fundamental company reports vary depending on the nature of the business. Examples of fundamental features include total revenue, gross profit, total assets, total debt, operating cash flow, and capital expenditure (3).

The sentimental analysis relates to the public's general feeling or attitude toward specific stocks as it relates to its success or failure within a given market (4). The goal of each of these methods is to try and predict market trends, giving investors the information necessary to productively place their money in a place where it will increase their overall investment (5).

In a survey of the types of analysis performed on over 300 samples, 66% of papers focused on technical analysis, 23% on fundamental analysis, and 11% based on some combination of the two or some form of sentimental analysis (2). Given the vast amount of research available, this paper will serve as a foundation for applying machine learning techniques to fundamental data analysis.

The uniqueness of this paper is in the focus of the combination of fundamental data classified based on the high and low technical indicators into three distinct classes: buy, sell, or hold. Along with the classification system, a broad

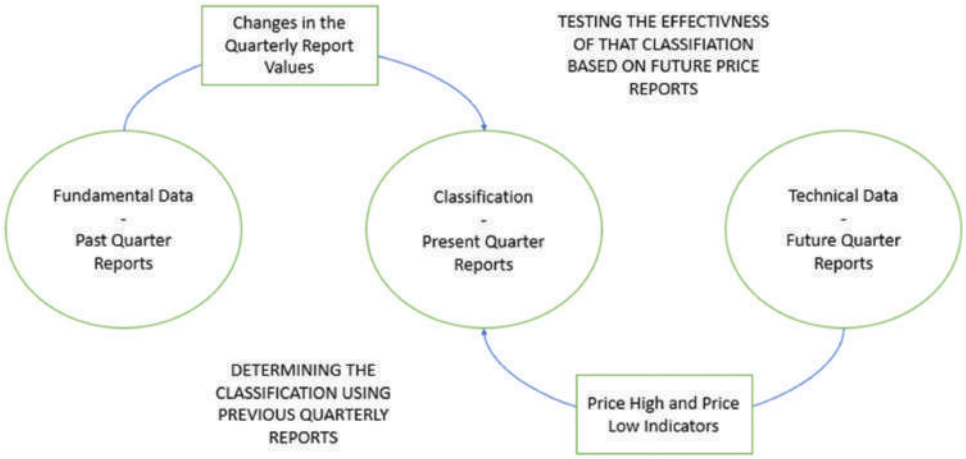

FIGURE 1 | Buy, sell, or hold classification process.

array of algorithms is applied in their most basic form, with the intended purpose of providing a benchmark performance of each classifier. The purpose of this is to gain useful insights into how these algorithms could be modified and expanded on for future use. This paper presents the results of collecting fundamental data on the top one hundred stocks in the NASDAQ stock market and applying eight different machine learning algorithms to predict whether a stock should be bought, sold, or held in any given quarter over the past 20 years from 2000 to 2020.

Most research focusing on technical analysis deals with, at its smallest, minute-to-minute prediction models Lamouirie and Achchab, (6), and at its largest, day-to-day (5). While useful, we intended to explore longer-term investment options that would be more useful to private citizens and long-term investors who wish to avoid the risk associated with day trading.

Given a large amount of research done on technical analysis and the generally positive results gained from that research, we began by drawing inspiration from Wang et al. (7), who attempted to train deep learning networks to analyze the Singapore Stock Exchange, straying from conventional trend studies to have their algorithms produce trading decisions directly. Their algorithms provided a buy, sell, or hold decision on a stock based on indicators gathered from a random forest algorithm. In 2017, Thakur et al. (8) repeated this method, expanding on using random forest algorithms to determine the rules used to classify each index as a buy/sell/hold index.

The purpose of this research is to allow non-investors a platform to study and enter the market, streamlining the results directly into a decision stating, that is, if a stock index should be bought, sold, or held. Discretizing the large number of fundamental features into a smaller number is a secondary focus of this study.

Hence, this study focuses on using fundamental values to produce decisions based on those same technical indicators. By associating the fundamental features with a decision based on the technical indicators, we have combined two methods of study, namely, technical and fundamental. We will study the fundamentals to predict classes based on the technical.

Given those articles and their influence on the work performed, it is prudent to note how this work will differ from these works. While many of the studies mentioned used machine learning algorithms (9–11), none used them on fundamental data to predict long-term results, which for the purpose of this paper is defined as results in increments of greater than 30 days. This project attempts to forecast the decision in 90-day increments four times a year, over a 20 year period, allowing personal non-day trading investors to use this information to invest responsibly and reliably in a volatile market environment.

By collecting quarterly data from 100 different stocks over a 20-year period, it is the work's commitment to relate fundamental data to predicted classification on the rise and fall of technical indicators and then produce a decision for the user to buy, sell, or hold a stock. The report will also explore which fundamental features gathered from the quarterly

TABLE 1 | Top 10 features selected from correlation and decision tree methodologies.

Correlation Method	Decision Tree Method
Basic Average Shares	Capital Expenditure
Diluted average shares	Total assets
Tax effect of unusual Items	End cash position
Other income expenses	Ordinary shares number
Total liabilities net minority Interest	Total liabilities net minority interest
Total unusual items	Total expenses
Excluding goodwill	
Total unusual items working capital	Reconciled depreciation gross profit
Total revenue operating expenses	Cost of revenue operating expenses

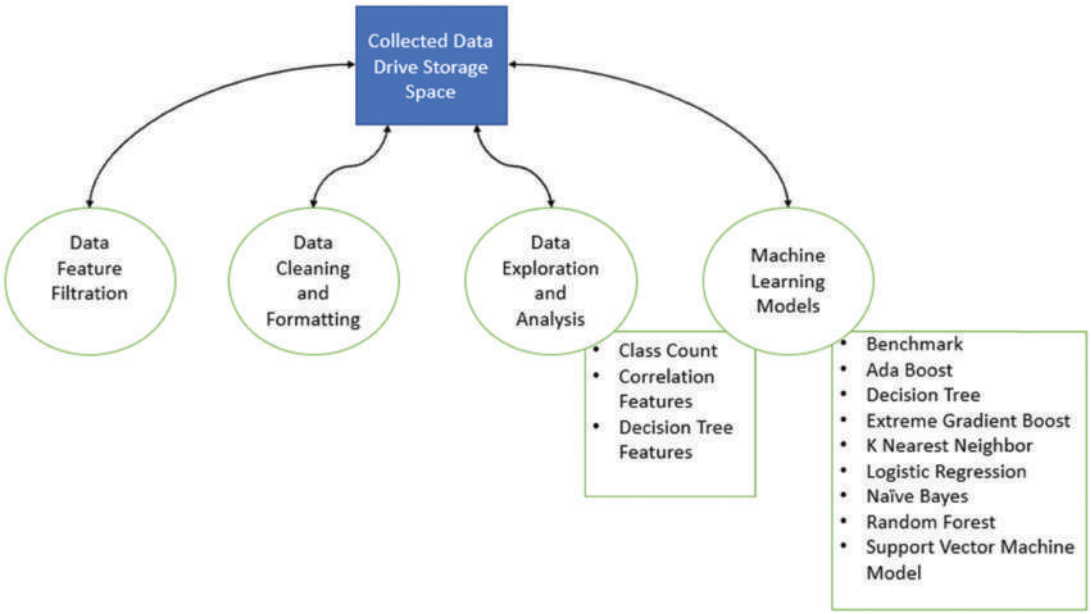

FIGURE 2 | Data processing and analysis framework.

report correlate the most with the decision classification, exploring two different methods of correlation and then producing two sets of features to be utilized in each of the machine learning algorithms. This project also serves the purpose of forming a benchmark foundation to continue studies in this field.

The remainder of this paper is organized as follows: Section "2. Data and Preprocessing" provides insight on the datasets utilized and the pre-processing performed to establish the final dataframe; Section "3. Algorithms" gives a high- level summary of the algorithms studied along with the parameters used in the experimentation; Section "4. Results" summarizes each algorithm's best parameters along with their results; and finally, Section "5. Conclusion and Future Works" points out the conclusions and posits future work to consider.

2. Data and preprocessing

The data for this project consists of all the data on the companies in the NASDAQ-100 stock market from January 1, 1999, to January 1, 2020, located in the Yahoo Finances database. The fundamental data are a collection of three separate reports pulled from the database. These dataframes (more technical term for files) and their feature counts were as follows: quarterly balance sheet (92), quarterly cash flow (72), and quarterly financials (52). A fourth dataframe on the historical daily values of each stock (technical indicators) was also pulled: historical prices indices (7). The combined total original feature count was 223.

The data are manually collected from Yahoo Finance. To prepare the data, a series of pre-processing steps were taken. First, extraneous features were removed from the

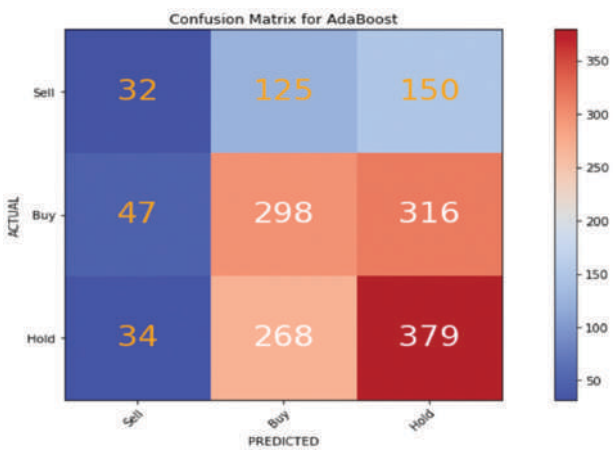

FIGURE 3 | Ada boost confusion matrix.

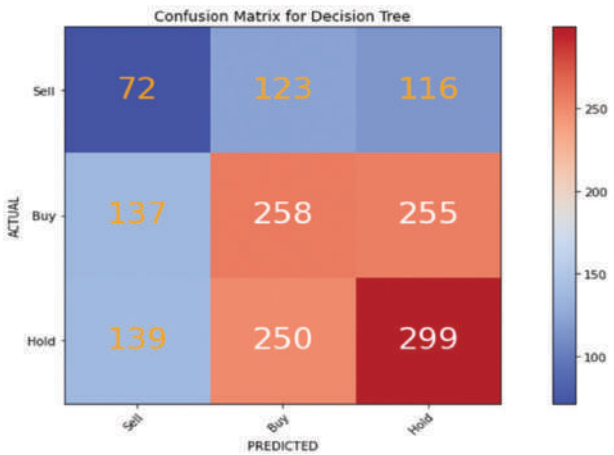

FIGURE 4 | Decision tree confusion matrix.

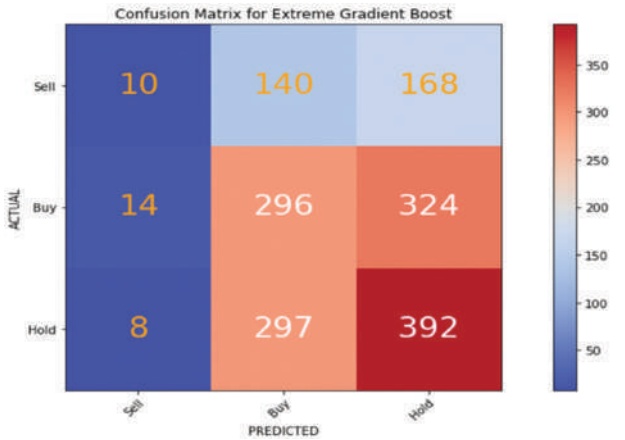

FIGURE 5 | Extreme gradient boost confusion matrix.

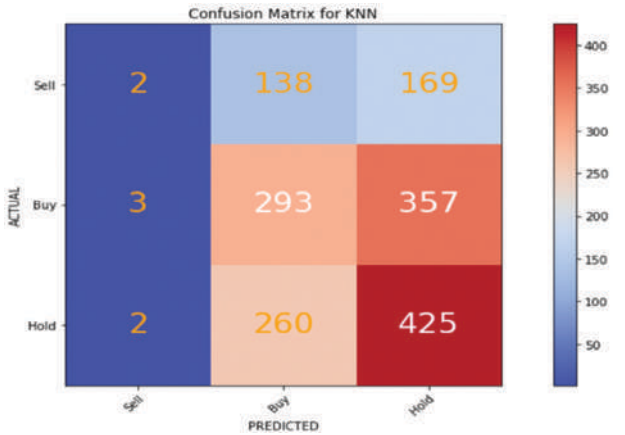

FIGURE 6 | K nearest neighbor confusion matrix.

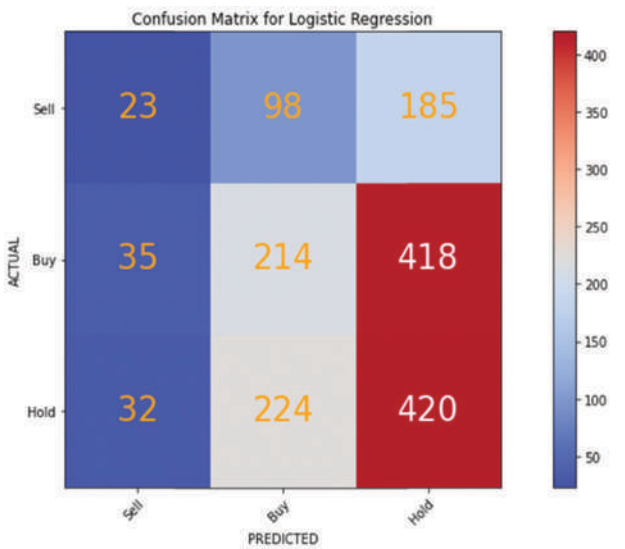

FIGURE 7 | Logistic regression confusion matrix.

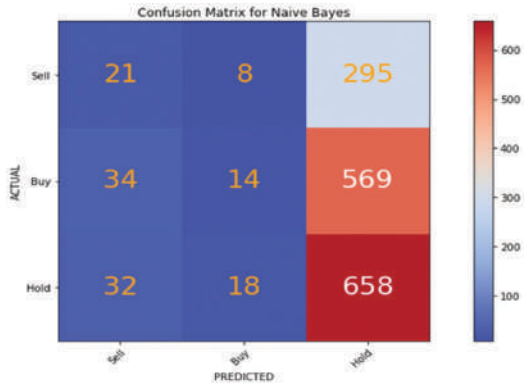

FIGURE 8 | Naive Bayes confusion matrix.

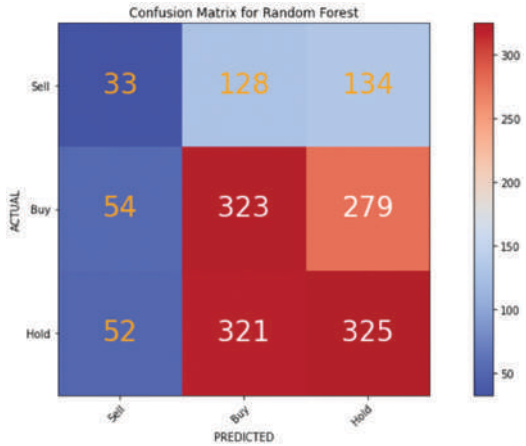

FIGURE 9 | Random forest confusion matrix.

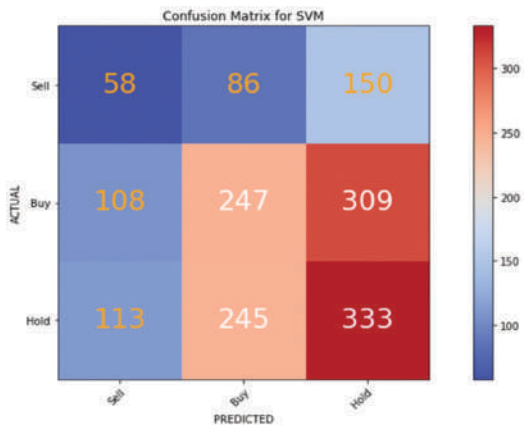

FIGURE 10 | Support vector machine confusion matrix.

dataframes. Many features were not reported continuously across all dataframes from each stock. Also, Yahoo Finance organizes features into individual sections and subsections, allowing for the generalizations of several features. Many of the subsections contained no values. Once the original dataframes were feature filtered, they were combined into a single quarterly report with the data associated around the dates that correlate with the end of the quarters of the fiscal calendars across all 20 years of data. This left us with a combined dataframe of the fundamental and technical

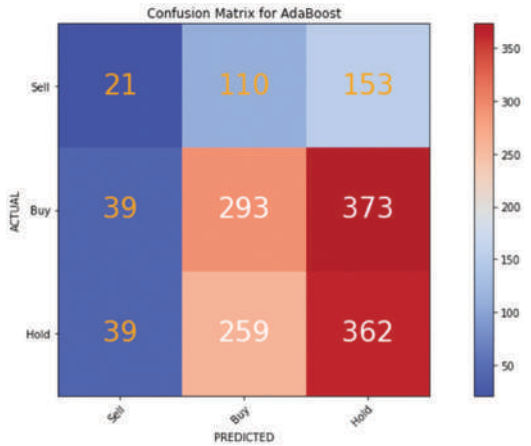

FIGURE 11 | AdaBoost confusion matrix.

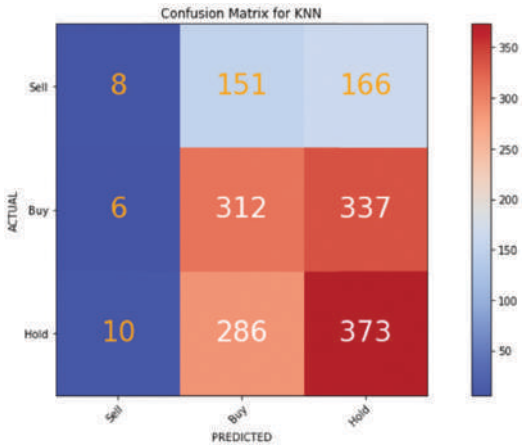

FIGURE 14 | K nearest neighbor confusion matrix.

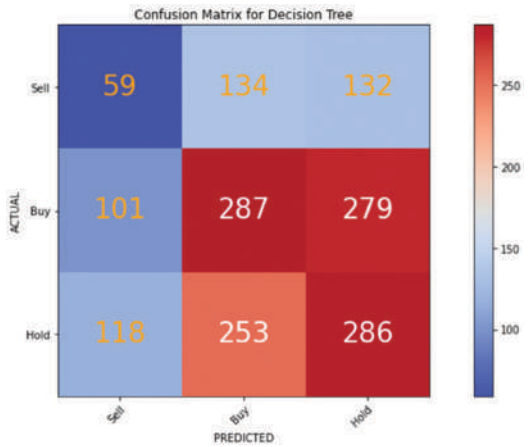

FIGURE 12 | Decision tree confusion matrix.

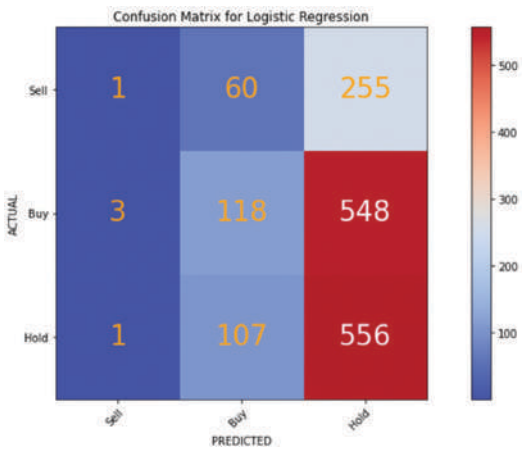

FIGURE 15 | Logistic regression confusion matrix.

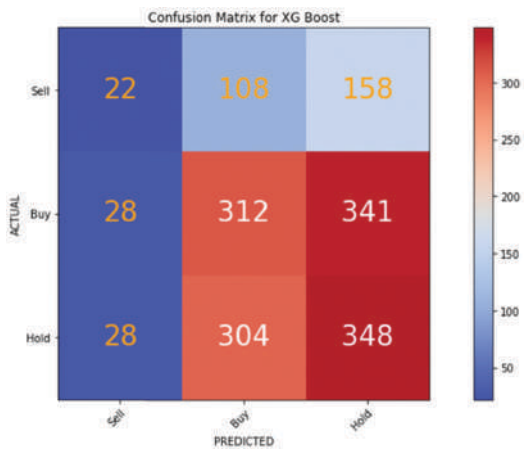

FIGURE 13 | Extreme gradient boost confusion matrix.

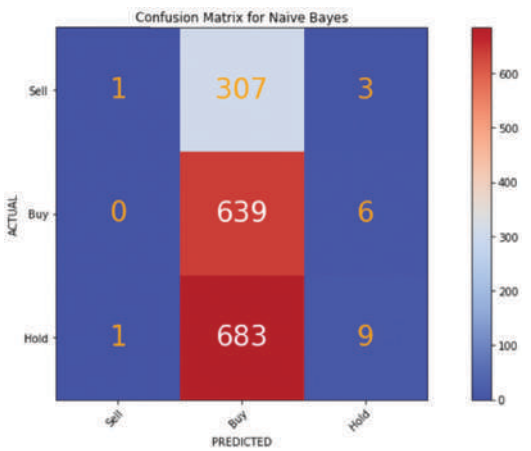

FIGURE 16 | Naive Bayes confusion matrix

values. The pre-processed dataframe consisted of 62 features of data and 8,498 indices of reported stock figures.

Next, the data were categorized. Each quarterly report was categorized into one of three possibilities: buy, sell, or hold if neither buy nor sell. The exact class necessities were as follows:

(1) Sell – High and low decrease by 5% or more in the next quarter.

(2) Buy – High and low increase by 5% or more in the next quarter.

(3) Hold – neither buy nor sell happens.

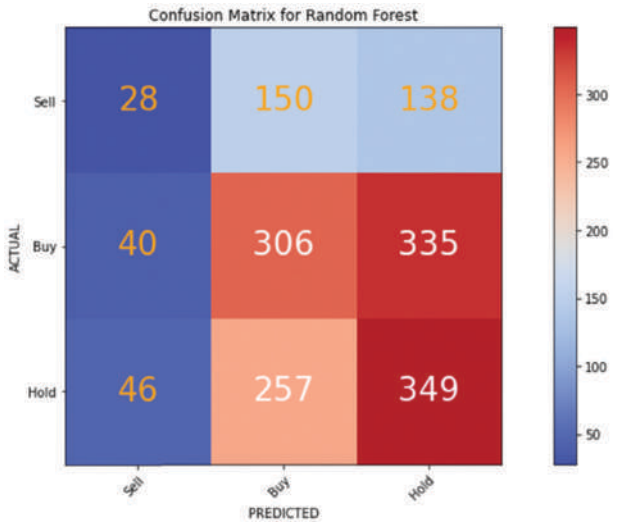

FIGURE 17 | Random forest confusion matrix.

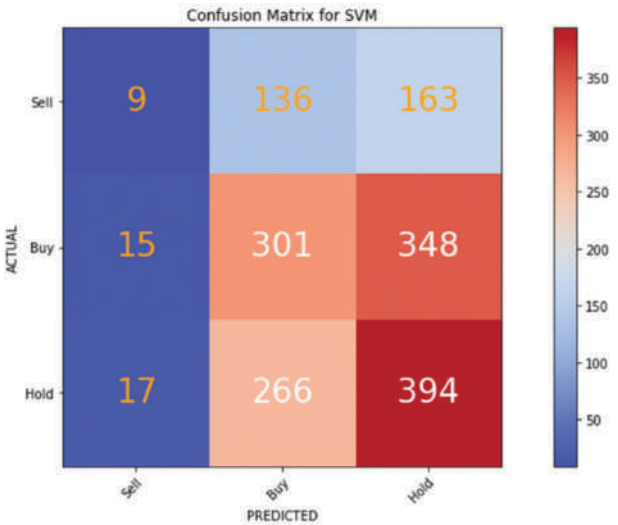

FIGURE 18 | Support vector machine confusion matrix.

Each index represents the quarterly reports and the high and low values associated with those quarters. These indices were classified based on the stated rules. Once the classifications were added to each company's quarterly reports, the rest of the data could be transitioned as follows. This methodology is demonstrated in **Figure 1**.

A categorization was decided upon. The values in the reports were altered to measure the percent change from the previous QR to the current QR and new classifications can be added. A feature was added to the dataframe for each existing feature. This new feature would measure the change in each feature from one quarter to the next. For example, say that in the previous quarter, a company was valued at $1,000, and in the next quarter, it was valued at $1,100. This represents an increase of 10%. The new feature replaced the price value of 1,100 with 10% for our current quarterly report. This process was repeated for every quarterly report, excluding the first, as no previous data existed with which to modify the data. This left every quarterly report with a percent change for each fundamental value.

In summary:

- All the data files were collected into a single dataframe for the purposes of preprocessing and exploring the data.
- To train and test the classifier, the price-related features were separated into two different dataframes with a similar index value. This was done to prevent data leakage during the training and testing of the model.
- The date indices were replaced with a simple number of indices as dates were no longer needed.

2.1. Feature select in using correlation values and tree classifiers

The fundamental data were collected, preprocessed, and reformatted. The final files were exported. The data exploration consisted of two separate but similar steps. Given the large number of features (59, after removing the name, symbol, and date columns), we wanted to condense the features into a more discreet number. The original features are presented in **Appendix Table A1**.

To perform this feature selection, we used two methodologies: correlation values and tree classifiers. For the first method, a correlation matrix was created, and the top 10 features were correlated with our decision classifications. Once the top 10 correlation features were located, the preprocessed dataframe was spliced to only include those features along with the decision classifications. The dataframe was exported for use in our models. Next, using SciKit, a decision tree was implemented to determine this set of top ten features to study. Once identified, they were also spliced out of the preprocessed dataframe and moved into a new dataframe along with the correlating classification labels. **Table 1** shows the features selected by both methods of feature discretization and used for the remainder of this experiment.

Using two different methods to determine which features to use will allow us to compare how the feature selection affects the accuracy of the models.

Now that the data were cleaned, formatted, and discretized features were selected, we finally began classifying the stocks using their quarterly reports. To do this, multiple different machine learning classifiers were used. The data were tested using eight different classifiers along with a dummy model to provide us with a benchmark to compare our results against.

- Ada Boost
- Decision Tree
- Extreme Gradient Boost
- K Nearest Neighbor

TABLE 2 | Ada boost algorithm synopsis.

Ada Boost - an estimator that initially fits on the original dataframe and then fits again on the same dataframe but in areas where the weights are incorrectly assigned those instances are re classified and more difficult instances become the focus.

Parameters

– n_estimators - The maximum number of estimators at which boosting is terminated. In case of a perfect fit, the learning procedure is stopped early.

  ○ 50, 100, 200, 500

– learning_rate - Weight applied to each classifier at each boosting iteration. A higher learning rate increases the contribution of each classifier. There is a trade-off between the learning rate and estimator parameters.

  ○ 1, 0.1, 0.01

Advantages

– Less prone to overfitting data

– Input parameters are not jointly optimized.

Disadvantages

– Requires quality dataset void of noisy and outlier data.

– Statistically slower compared to other algorithms

TABLE 3 | Decision tree algorithm synopsis.

Decision Tree - Uses a tree data structure to predict the results of a particular classification. Highly useful classification model.

Parameters

– *criterion* - defines the function used to measure the quality of a split.

  ○ 'gini' and 'entropy'

– *max_depth* - defines the max depth of the tree. If 'none' nodes are expanded pure

  ○ None, 2, 3, 4, 5, 6

– *min_samples_split* - defines the min number of samples required to split a node

  ○ 2, 5, 10

– *min_samples_leaf* - defies the min number of samples required at a leaf node to split it.

  ○ 1,2,3,4,5,6

Advantages

– Easy to understand and implement

– Insensitive to missing values

– Uncorrelated features can be processed with positive correlated.

results.

Disadvantages

– Multiclassification problems increase error rates

– Underperforms when multiple features are highly

TABLE 4 | Gradient boost algorithm synopsis.

Extreme Gradient Boost - In each stage, n class regression trees are fit to the negative gradient of a multinomial deviance loss function which allows for the enhancement of arbitrary differentiable loss functions. Essentially each model is trained on the failures of the previous model.

Parameters

– Booster – which booster to use.

  ○ "gbtree", "gblinear", "dart"

– Eta – step size shrink value used to prevent overfitting.

  ○ 0.1,0.5,0.9

– Gamma – Minimum loss reduction required to make a further partition on a leaf node of the tree. The larger the gamma is, the more conservative the algorithm will be.

  ○ 0, 1, 3

– n_estimators – number of trees in the forest

  ○ 50, 100, 200

– max_depth – The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure

  ○ 1, 3, 6

Advantages

– Efficient classification model

– Historically more accurate then random forest

– Can handle mixed feature types.

Disadvantages

– Sensitive to outliers due to the carry through of errors in previous iterations

– Difficult to upscale because of its reliance on previous iterations.

TABLE 5 | K nearest neighbor algorithm synopsis.

K Nearest Neighbor – A supervised machine learning algorithm that finds the distances between all the examples in the data by selecting K closest examples. Chosen due to the high relation between two close data points in our data set.

Parameters

n_neighbors – number of neighbors to use.

- 50, 100, 200

weights – the weight function used in prediction.

- 'uniform' – all points in each neighborhood are equally weighted.
- 'distance' – closer neighbors on a query point will have more influence.

p – the parameter for the Minkowski metric

- 1 – equivalent to Manhattan distance
- 2 – uses the Euclidean metric

Advantages

– Versatile algorithm that can be used for classification, regression, and search

Disadvantages

– Speed in directly related to the size of the data making this classifier hard to size up.

TABLE 6 | Logistic regression algorithm synopsis.

Logistic Regression

– Used to assign observations to a discrete set of classes using a predictive analysis algorithm based on probability calculated using a sigmoid cost function.

Parameters

– penalty – specify the norm of the penalty

- l1 - add a l1 penalty term
- l2 - add an l2 penalty term

– fit_intercept – specifies if a constant should be added to the decision function

- True, False

– intercept_scaling – used when using liblinear parameter and True self.fit interceptor it lessens the effect of regular synthetic weights.

- 1, 10, 50

– Solver – chose the algorithm used in the optimization problem

- 'liblinear' – one vs rest schema
- 'saga' – used for larger dataframes to handle multinomial loss

Advantages

– Performs well with continuous or categorical data.

– Easy to use a interpret the results

– Feature scaling not needed

Disadvantages

– Data intensive

– Sensitive to multi-collinearity

– Performs poorly with non-linear data

– Prone to overfitting the data

- Logistic Regression
- Naive Bayes
- Random Forest
- Support Vector Machine Model
- Dummy (Benchmark)

Each of these models were trained and fitted to the dataset to determine the best performing model. Along with running the default algorithms, we will also perform a grid search on several different parameters to try and identify the best results for each algorithm. **Figure 2** lays out this entire process.

Results are presented in terms of four different statistical metrics: accuracy, precision, recall, and F1-score. Our pre-processing methodology produced a slightly imbalanced dataset, hence this study places more importance on precision since it deals with the amount of false positives. When studying stocks, we have chosen to be conservative with our investing policy. Focusing on precision allows us to avoid investing in the wrong stock over recall, which would focus our concern on missing the opportunity. Each of the values has its importance, but because we do not want to invest in a stock that is a sell, we will focus on precision. The confusion matrix for each of the algorithms is also included to picture how each of our models predicted vs. the actual breakdown of how each index was classified. The complete results and diagrams for each set of features can be found in **Tables 10**, **11** and **Figures 3–18**.

TABLE 7 | Naive Bayes algorithm synopsis.

Naive Bayes – A supervised learning algorithm used for classification by features assuming each feature is independent of each other with no correlation.

Parameters

– var_smoothing – Portion of the largest variance of all features that is added to variances for calculation stability.

  ○ 1.5**-i for i in range (–20, 20, 2)

Advantages	***Disadvantages***
– Fast paced algorithm that can be used in real time	– Assumes each feature make an equal contribution,
– Scalable to larger datasets weighs each feature equally.	– Requires each classification to be well represented.
– Good performance with high dimensional data	

TABLE 8 | Random forest table synopsis.

Random Forest - Using many individual decision trees, each of which returns a class prediction and the class with the most returns becomes the model's prediction.

Parameters

– n_estimators – number of trees in the forest

  ○ 10,50,100,200

– criterion – the function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain.

  ○ 'gini' and 'entropy'

– max_depth – The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure

  ○ None, 2, 5, 10

– min_samples_split – The minimum number of samples required to split an internal node

  ○ 5, 10

– min_samples_leaf – The minimum number of samples required to be at a leaf node to be considered to continue splitting.

  ○ 1, 2, 5

Advantages	***Disadvantages***
– Works well with unbalanced data.	– Smaller data frames and low dimension data are prone to in accurate classifications.
– Excellent non–linear classifier.	
– Maintains high accuracy when used with data that has missing values.	– Setting parameters is difficult and sometimes randomized.

TABLE 9 | Support vector machine model algorithm synopsis.

Support Vector Machine Model - An extension of the maximal margin classifier modified for general use cases especially non-linear features.

Parameters

- C - Regularization parameter. The strength of the regularization is inversely proportional to C.

  ○ [0.01,0.1,1],

- kernel is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape 'rbf', 'sigmoid', 'linear'

Advantages	***Disadvantages***
– Works well in high dimensional spaces where the dimensions is greater than the data frames.	– Better suited for binary classifications.
– Avoids overfitting the data due to outliers.	– Performs slower on larger datasets.
	– Selecting the right kernel function is difficult and can be random.

3. Algorithms

Tables 2–9 give a short overview of the algorithms used in this study, their advantages and disadvantages, as well as any parameter sets for each model. A total of eight algorithms were chosen based on their use in previous studies on stock market data.

Each algorithm was tested several times, using all of the default methodologies as well as altering specific parameters using a grid search technique. This was done to ensure that we were locating the optimal settings for each model and so that we could in turn find the optimal model. This may increase the processing time taken because each model will need to be run for each combination of parameters but eventually produces better results.

TABLE 10 | Results of top features based on a tree classifier.

Decision Tree Algorithms Results						
Algorithms	**Class**	**Precision**	**Recall**	**F1-score**	**Accuracy**	**Parameters Used**
AdaBoost	Sell	0.28	0.10	0.15	0.43	learning_rate = 1
	Buy	0.43	0.45	0.44		n_estimators = 500
	Hold	0.45	0.56	0.50		
Decision	Sell	0.21	0.23	0.22	0.38	criterion = 'entropy'
Tree	Buy	0.41	0.40	0.40		max_depth = None
	Hold	0.45	0.43	0.44		min_samples_leaf' = 1 min_samples_split = 5
Extreme	Sell	0.31	0.03	0.06	0.42	Booster = gbtree
Gradient	Buy	0.40	0.47	0.43		eta = 0.1
Boost	Hold	0.44	0.56	0.50		gamma = 0 grow_policy = depthwise max_depth = 6
						n_estimators = 50
K Nearest	Sell	0.29	0.01	0.01	0.44	n_neighbors = 50
Neighbor	Buy	0.42	0.45	0.44		p = 2
	Hold	0.45	0.62	0.52		weights = 'distance'
Logistic	Sell	0.22	0.03	0.05	0.43	C = 1.0
Regression	Buy	0.43	0.33	0.37		fit_intercept = False
	Hold	0.43	0.70	0.54		intercept_scaling = 1 penalty = 'l2' solver = liblinear
Naive	Sell	0.24	0.06	0.10	0.42	var_smoothing
Bayes	Buy	0.35	0.02	0.04		= 0.001522438
	Hold	0.43	0.93	0.59		8403474447
Random	Sell	0.34	0.08	0.14	0.47	criterion = 'gini'
Forest	Buy	0.42	0.48	0.45		max_depth = None
	Hold	0.46	0.54	0.50		min_samples_leaf = 1 min_samples_split = 5
						n_estimators = 10
Support	Sell	0.21	0.20	0.20	0.39	C = 1
Vector	Buy	0.43	0.37	0.40		Kernel = sigmoid
Machine	Hold	0.42	0.48	0.45		
Benchmark	Sell	0.15	0.15	0.15	0.34	N/A
Model	Buy	0.26	0.26	0.27		
	Hold	0.27	0.24	0.29		

The parameters tested are listed along with their algorithm's synopsis and a short description of what the parameter effects are. Each of the two discrete top features will both be tested in this manner, choosing the best set of parameters, which will in turn find the best algorithm for each set of features. Of the parameters listed for each algorithm, when more than one parameter value is listed, each listed parameter was tested for that model and the specific combination of parameters that produced the best results of all attempts for each model. The definitions for each parameter were taken from Pedregosa et al. (12) and the SciKit learn documentation. The dummy algorithm was run using Scikit Learn's default classifier.

4. Results

The results displayed in **Tables 10**, **11** show what each algorithm returned using the best tuned parameters found through the grid search phase of the experiment. The results are reported on four different values: precision, recall, the f1-score for each classification, and the overall accuracy of each model. Each metric is reported for each of the three classifications (buy, sell, and hold). Due to the imbalanced nature of the classifications, we prioritize precision over accuracy when determining the efficacy of the classifiers. Along with the results in both tables, the confusion matrix for each model discretization method combination is presented in **Figures 3–18**. The confusion matrix allows us to visualize in depth how each model performed by displaying the

TABLE 11 | Results of top features based on a correlation model.

Correlation Method Algorithms Results						
Algorithms	**Class**	**Precision**	**Recall**	**F1-score**	**Accuracy**	**Parameters Used**
AdaBoost	Sell	0.27	0.06	0.10	0.41	learning_rate = 1
	Buy	0.40	0.46	0.43		n_estimators = 500
	Hold	0.43	0.55	0.48		
Decision	Sell	0.16	0.17	0.16	0.37	Criterion = entropy
Tree	Buy	0.41	0.42	0.42		max_depth = None
	Hold	0.44	0.42	0.43		min_samples_leaf = 6 min_samples_split = 5
Extreme	Sell	0.28	0.22	0.22	0.41	Booster = 'gbtree
Gradient	Buy	0.43	0.46	0.44		eta = 0.1
Boost	Hold	0.44	0.51	0.46		gamma = 0 grow_policy = depthwise max_depth = 6, n_estimators = 200
K Nearest	Sell	0.32	0.04	0.07	0.44	n_neighbors = 50
Neighbor	Buy	0.43	0.45	0.44		p = 2
	Hold	0.42	0.57	0.48		weights = 'distance'
Logistic	Sell	0.22	0.03	0.05	0.43	C = 11.390625
Regression	Buy	0.43	0.33	0.37		fit_intercept = False
	Hold	0.43	0.70	0.54		intercept_scaling = 1 penalty = 'l1' solver = 'liblinear '
Naive	Sell	0.08	0.00	0.01	0.40	var_smoothing
Bayes	Buy	0.40	0.97	0.57		= 0.0077073466292589396
	Hold	0.52	0.02	0.05		
Random	Sell	0.27	0.10	0.15	0.42	Criterion = 'entropy'
Forest	Buy	0.45	0.46	0.45		max_depth = None
	Hold	0.42	0.52	0.46		min_samples_leaf = 2 min_samples_split = 5 n_estimators = 10
Support	Sell	0.21	0.10	0.14	0.40	C = 1
Vector	Buy	0.43	0.37	0.40		Kernel = 'sigmoid'
Machine	Hold	0.41	0.57	0.48		
Benchmark	Sell	0.15	0.15	0.15	0.34	N/A
Model	Buy	0.26	0.26	0.27		
	Hold	0.27	0.24	0.29		

number of indices that were misclassified for each of the three classifications. By breaking down each classification into true classifications and false classifications and also labeling how each false classification was mislabeled, we can gain a deeper understanding how the algorithms performed and form a foundation for improvement.

When viewing the confusion matrices for each discretization method and focusing on the top performing algorithms, we can see that for both methodologies, the results were best at predicting true holds, then true buys, and rarely correctly predicted true sells. This indicates that the weight was placed on being more conservative, leaning toward holding over buying and selling. While these decisions are not straightforward, the most important aspect was for the algorithms to correctly classify buy and hold indices over incorrectly sold ones, as buying and holding are more directly related to money lost (i.e., buying a stock that is going to lose value or holding a stock that will lose value both cost you money you already have, whereas selling a stock that will gain you money costs you potential income you have not yet gained).

4.1. Top 10 features based on a decision

4.1.1. Tree classifier

The top 10 features are presented in **Table 1**.

TABLE 12 | Result from most optimal runs of both discretization models.

Best Performing Algorithms from Both Methodologies						
Algorithms	**Class**	**Precision**	**Recall**	**F1-score**	**Accuracy**	**Parameters Used**
K Nearest Neighbor (Correlation Method) Sell	0.32	0.04	0.07	0.44	n_neighbors = 50	
	Buy	0.43	0.45	0.44		p = 2
	Hold	0.42	0.57	0.48		weights = 'distance'
Random Forest (Tree Method)	Sell	0.34	0.08	0.14	0.47	criterion = 'gini'
	Buy	0.42	0.48	0.45		max_depth = None
	Hold	0.46	0.54	0.50		min_samples_leaf = 1 min_samples_split = 5 n_estimators = 10
Benchmark Model	Sell	0.15	0.15	0.15	0.34	N/A
	Buy	0.26	0.26	0.27		
	Hold	0.27	0.24	0.29		

4.2. Summary of top performing algorithms from each methodology

Table 12 displays the best results from both the discretization methods applied. As can be seen below, the random forest nodel has an increase in accuracy of 13% while the K nearest neighbor has an increase in accuracy of 10%. Along with a dramatic increase in accuracy, there is also a dramatic increase in the precision across all of the classifications, nearly doubling the sell classification and over a 15% increase in both the buy and hold classifications.

5. Conclusion and future works

By collecting quarterly data from 100 different stocks over a 20-year period, it is the work's commitment to relate fundamental data to predicted classification on the rise and fall of technical indicators and then produce a decision for the user to buy, sell, or hold a stock. The report will also explore which fundamental features gathered from the quarterly report correlate the most with the decision classification, exploring two different methods of correlation and then producing two sets of features to be utilized in each of the machine learning algorithms. This project also serves the purpose of forming a benchmark foundation to continue studies in this field.

Based on this work, it can be concluded that when continuing this line of study, any efforts should be focused on the Knearest neighbor and the random forest algorithms as they showed the best improvement against the benchmark model. It should also be noted that, while the percentages could be considered low, given the nature of our study, the ability of our classifiers to predict the highest reported precision of 46% and accuracy of47%should be considered a significant improvement. Given the unforgiving nature of the study due to the volatile and unpredictable nature of the data, more work need to be done in this area, but this study shows that fundamental analysis at this stage forms a foundation for future studies. It can also be noted that, on average, the decision tree algorithm results were better than the correlation-based algorithm results. Also, it can be noted that, on average, the precision of the sell was lower than the precision of the buy or hold.

For future work, we are thinking along the lines of: (i) First and foremost, expanding our dataset to all the available stock indices in the Yahoo Finance database and forming a data pipeline to potentially allow our data to be used indefinitely as new data is produced and posted; (ii) combining the best performing algorithms to increase the performance of our models; and finally, (iii) exploring the effect of modifying the features by creating interactive features using domain knowledge.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Author contributions

NM was responsible for the initial research and study, including the collecting of related works, performing the study of machine learning algorithms, and the initial draft of the manuscript. He also contributed to the final submission. SB was responsible for guiding NM as he conducted his research and advising on the research topic and formation. She also coauthored and edited the final submission. Both authors contributed to the article and approved the submitted version.

References

1. Ayala J, Garda-Torres M, Noguera JLV, Gomez-Vela F, Divina F. Technical analysis strategy optimization using a machine learning approach in stock market indices. *Knowl Based Syst.* (2021) 225:107119. doi: 10.1016/j.knosys.2021.107119
2. Cohen G, Kudryavtsev A, Hon-Snir S. Stock market analysis in practice: is it technical or fundamental? *J Appl Finan Bank.* (2011) 1:125–38.
3. Nti IK, Adekoya AF, Weyori BA. A systematic review of fundamental and technical analysis of stock market predictions. *Artif Intell Rev.* (2019) 53:3007–57. doi: 10.1007/s10462-019-09754-z
4. Mizuno T, Ohnishi T, Watanabe T. Novel and topical business news and their impact on stock market activity. *EPJ Data Sci.* (2017) 6:1–14. doi: 10.1140/epjds/s13688-017-0123-7
5. Chong E, Han C, Park FC. Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. *Expert Syst Applic.* (2017) 83:187–205. doi: 10.1016/j.eswa.2017.04.030
6. Lanbouri Z, Achchab S. Stock market prediction on high frequency data using long-short term memory. *Procedia Comp Sci.* (2020) 175:603–8. doi: 10.1016/j.procs.2020.07.087
7. Wang Q, Li J, Qin Q, Sam Ge S. Linear, adaptive and nonlinear trading models for Singapore stock market with random forests. *Proceedings of the 2011 9th IEEE International Conference on Control and Automation (ICCA)*. Paris (2011).
8. Thakur M, Kumar D. A hybrid financial trading support system using multi-category classifiers and random forest. *Appl Soft Comput.* (2018) 67:337–49. doi: 10.1016/j.asoc.2018.03.006
9. Patel J, Shah S, Thakkar P, Kotecha K. Predicting stock market index using fusion of machine learning techniques. *Expert Syst Applic.* (2015) 42:2162–72. doi: 10.1016/j.eswa.2014.10.031
10. Basak S, Kar S, Saha S, Khaidem L, Dey SR. Predicting the direction of stock market prices using tree-based classifiers. *North Am J Econ Finan.* (2019) 47:552–67. doi: 10.1016/j.najef.2018.06.013
11. Vijh M, Chandola D, Tikkiwal VA, Kumar A. Stock closing price prediction using machine learning techniques. *Proced Comp Sci.* (2020) 167:599–606.
12. Pedregosa F, Duchesnay E, Perrot M, Brucher M, Cournapeau D, Passos A, et al. Scikit-learn: machine learning in python. API design for machine learning software: experiences from the scikit-learn project. *J Mach Learn Res.* (2011) 12:2825–30.
13. Picasso A, Merello S, Ma Y, Oneto L, Cambria E. Technical analysis and sentiment embeddings for market trend prediction. *Expert Syst Applic.* (2019) 135:60–70. doi: 10.1 016/j.eswa.2019.06.014
14. Dai Y, Shang Y. *Machine learning in stock price trend forecasting.* Stanford: University of Stanford (2013).
15. Kulshreshtha S, Vijayalakshmi A. An ARIMA- LSTM hybrid model for stock market prediction using live data. *J Eng Sci Technol Rev.* (2020) 13:117–23. doi: 10.25103/jestr.1 34.11
16. Jahufer A. Choosing the best performing garch model for Sri Lanka stock market by non-parametric specification test. *J Data Sci.* (2021) 13:457–72.
17. Jeric SV. Rule extraction from random forest for intra-day trading using crobex data. *Proc FEB Zagreb Int Odyssey Conf Econ Bus.* (2020) 2:411–9.
18. Liu LJ, Shen W-K, Zhu J-M. Research on risk identification system based on random forest algorithm-high-order moment model. *Complexity* (2021) 2021:1–10.
19. Chen M, Zhang Z, Shen J, Deng Z, He J, Huang S. A quantitative investment model based on random forest and sentiment analysis. *J. Phys Conf Ser.* (2020) 1575:12083. doi: 10.1088/1742-6596/1575/1/012083
20. Ciner C. Do industry returns predict the stock market? A reprise using the random forest. *Q Rev Econ Finan.* (2019) 72:152–8. doi: 10.1016/j.qref.2018.11.001
21. Patel J, Shah S, Thakkar P, Kotecha K. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Syst Applic.* (2015) 42:259–68. doi: 10.1016/j.eswa.2014 07.040
22. Yahoo. *Yahoo Finance - Stock Market Live, quotes, Business & Finance News. Yahoo! Finance*. Sunnyvale, CA: Yahoo (2022).

Appendix

TABLE A1 | Original features

Features	Description	Range of Values
Date	Date that each value was reported	Dates from 12/31/1999 to 01/01/2020
Name	Name of each stock as reported in NASDAQ 100	String values of varying lengths
Symbol	Symbol used to associate each stock to its name within stock market	Strings values from three to four characters.
DilutedAverage Shares	**Shares outstanding after all conversion possibilities are**	**Dollar values from 9,999,999,999 - 9,999,999,1013**
TotalOperating IncomeAsReported	implemented Sum total of profit after subtracting regular, recurring costs and	Dollar values from 9,999,999,999 - 9,999,999,1014
TotalExpenses+	expenses Sum of cost of sales and operating expenses	Dollar values from 9,999,999,999 -
NetIncomeFrom Continuing and DiscontinuedOperation	After-tax earnings generated	9,999,999,1015 Dollar values from 9,999,999,999 - 9,999,999,1016
NormalizedIncome	Clearing impact of non-recurring items	Dollar values from 9,999,999,999 -
InterestIncome	Taxable income	9,999,999,1017 Dollar values from
InterestExpense	Cost of borrowing money from banks, bond investors, and other	9,999,999,999 - 9,999,999,1018 Dollar values from 9,999,999,999 - 9,999,999,1019
NetInterestIncome	sources Difference between revenue from interest-bearing assets	Dollar values from 9,999,999,999 - 9,999,999,1020
EBIT	and expenses on interest-bearing liabilities Earnings before interest and taxes	Dollar values from 9,999,999,999 -
EBITDA	Earnings before interest, taxes, depreciation, and amortization	9,999,999,1021 Dollar values from 9,999,999,999 - 9,999,999,1022
ReconciledCostOfRevenue	Act of reconciling all sales	Dollar values from 9,999,999,999 -
Reconciled Depreciation + NetIncomeFrom	Fixed asset reconciliation statement Net income obtained	9,999,999,1023 Dollar values from 9,999,999,999 - 9,999,999,1024 Dollar values from
Continuing Operation	from net of minority share-holders	9,999,999,999 - 9,999,999,1025
NetMinority Interest TotalUnusual Items Excluding	Non-recurring gain or loss not considered part	Dollar values from 9,999,999,999 -
Goodwill*	of normal business	9,999,999,1026
TotalUnusualItems*	Non-recurring gains or losses not considered part of normal business	Dollar values from 9,999,999,999 - 9,999,999,1027
NormalizedEBITDA	Net income from continuing operations before interest, income taxes, depreciation and amortization, excluding any non-recurring items and/or non-cash equity compensation expense	Dollar values from 9,999,999,999 - 9,999,999,1028
TotalRevenue*	Sum of both operating and non-operating revenues of company as reported for any given quarter	Dollar values from 9,999,999,999 - 9,999,999,999
CostOfRevenue +	Cost of manufacturing and delivering product or service	Dollar values from -9.999.999.1000
GrossProfit+	Profit after deducting costs associated with making and selling products and/or providing services	Dollar values from -9,999,999,999 - 9,999,999,1001
OperatingExpense*+	Expense business incurs through its normal business operations	Dollar values from 9,999,999,999 - 9,999,999,1002
OperatingIncome	Profit realized from operations after deducting operating expenses	Dollar values from 9,999,999,999 - 9,999,999,1003
NetNonOperating InterestIncomeEx-	Expense unrelated to core operations;	Dollar values from 9,999,999,999 -

(Continued)

TABLE A1 | *(Continued)*

(Continued)

TABLE A1 | *(Continued)*

Features	Description	Range of Values
pense	Interest charged on loss of an asset; Does not include day to day expenses	9,999,999,1004
OtherIncome Expense*	Income that does not relate directly to business operations	Dollar values from 9,999,999,999 - 9,999,999,1005
PretaxIncome	Net sales minus cost of goods sold minus operating expenses	Dollar values from 9,999,999,999 - 9,999,999,1006
TaxProvision	Estimated income tax company is legally expected to pay	Dollar values from 9,999,999,999 - 9,999,999,1007
NetIncome Common-monStockholders	Bottom line profit belonging to common stockholders	Dollar values from 9,999,999,999 - 9,999,999,1008
DilutedNIAvailto ComStockholders	Diluted Net Income; net income adjusted for not paying out any interest expense or preferred dividends.	Dollar values from 9,999,999,999 - 9,999,999,1009
BasicEPS	Net income minus preferred dividends divided by weight average of common shares outstanding	Dollar values from 9,999,999,999 - 9,999,999,1010
DilutedEPS	Value used to gauge quality of earnings per share of stock	Dollar values from 9,999,999,999 - 9,999,999,1011
BasicAverageShares*	Average number of shares investors held at any point in period	Dollar values from 9,999,999,999 - 9,999,999,1012
TaxRateForCalcs	Effective federal tax rate	Dollar values from 9,999,999,999 - 9,999,999,1029
TaxEffectOf Unusualitems*	Net value of taxable unusual items	Dollar values from 9,999,999,999 - 9,999,999,1030
TotalAssets^{+}	Combined value of the total liabilities and shareholder 's equity	Dollar values from 9,999,999,999 - 9,999,999,1031
TotalLiabilities NetMinority Interest*$^{+}$	Share of equity ownership not owned or controlled by parent corporation	Dollar values from 9,999,999,999 - 9,999,999,1032
TotalEquityGross MinorityInterest	Minority Interests divided by the total equity	Dollar values from 9,999,999,999 - 9,999,999,1033
TotalCapitalization	Sum of the long-term debt and all other equities including common stock and preferred stock	Dollar values from 9,999,999,999 - 9,999,999,1034

(Continued)

TABLE A1 | *(Continued)*

Features	Description	Range of Values
CommonStockEquity	Stock held by founders and employees not included in stock owned by parent company	Dollar values from 9,999,999,999 - 9,999,999,1035
NetTangibleAssets	Total assets of company minus any intangible assets	Dollar values from 9,999,999,999 - 9,999,999,1036
WorkingCapital*	Capital used in day to day trading operations	Dollar values from 9,999,999,999 - 9,999,999,1037
InvestedCapital	Money raised by issuing securities, stock equity shareholders and debt of bond holders	Dollar values from 9,999,999,999 - 9,999,999,1038
TangibleBookValue	Book value	Dollar values from 9,999,999,999 - 9,999,999,1039
TotalDebt	Sum of short- and long-term debt	Dollar values from 9,999,999,999 - 9,999,999,1040
ShareIssued	Authorized shares sold to and held by shareholders of company	Dollar values from 9,999,999,999 - 9,999,999,1041
OrdinaryShares Number^{+}	Stocks sold on a public exchange.	Dollar values from 9,999,999,999 - 9,999,999,1042
OperatingCashFlow	Cash generated by normal business operation	Dollar values from 9,999,999,999 - 9,999,999,1043
InvestingCashFlow	Cash generated (or spent) on non-current assets	Dollar values from 9,999,999,999 - 9,999,999,1044
FinancingCashFlow	Generated cash flow to pay back loan	Dollar values from 9,999,999,999 - 9,999,999,1045
EndCashPosition^{+}	Cash on books at specific point in time	Dollar values from 9,999,999,999 - 9,999,999,1046
CapitalExpenditure^{+}	Used to undertake new projects or investments	Dollar values from 9,999,999,999 - 9,999,999,1047
IssuanceOf CapitalStock	**Amount of money** generated when company initially sold its common stock on open market	**Dollar values from** 9,999,999,999 - 9,999,999,1048
RepaymentOfDebt	After all long-term debt instrument obligations are repaid, balance sheet will reflect a canceling of principal and liability expenses for total amount of interest	Dollar values from 9,999,999,999 - 9,999,999,1049
RepurchaseOf	When a company buys	Dollar values from

(Continued)

TABLE A1 | *(Continued)*

Features	Description	Range of Values
CapitalStock	back its shares from marketplace	9,999,999,999 - 9,999,999,1050
FreeCashFlow	Cash generated after accounting for cash outflows	Dollar values from 9,999,999,999 - 9,999,999,1051
Open	Price at which financial security opens in market	Value from 0 to 100
High	Price at which financial security is highest on market	Value from 0 to 101
Low	Lowest price of financial security	Value from 0 to 102
Close	Closing price of financial security	Value from 0 to 103
Adj Close	Amends stock's closing price	Value from 0 to 104
Volume	Amount of asset or security that changes hands	Value from 0 to 999,999,999

* = Feature derived from Correlation Method.
+ = Feature derived from Decision Tree Method.
* + = Feature derived from both methodologies.
Please note that definitions of each feature were pulled from investopia or yahoo finance.