2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Stock Price Movement Prediction based on Optimized Traditional Machine Learning Models

José Junior de Oliveira Silva*, Roberto Souto Maior de Barros*, Silas Garrido Teixeira de Carvalho Santos*†
**Centro de Informática, Universidade Federal de Pernambuco*, 50740-560, Recife-PE, Brazil
{jjos,roberto,sgtcs}@cin.ufpe.br
†*SiDi Institute*, 51110-160, Recife-PE, Brazil
s.garrido@sidi.org.br

***Abstract*—Stock price prediction has attracted several investors willing to maximize their profits, believing the opportunities to expand their earnings are higher than using conventional financial approaches, such as savings or fixed deposits. Market analysts, traders, and researchers have investigated different techniques such as Bayesian model, Fuzzy classifiers, Artificial Neural Networks (ANN), Support Vector Machines (SVM), etc. to analyze stock markets and make trading decisions. More recently, deep learning models have gained prominence. However, because of the large amount of data required for training, these techniques typically aggregate all stocks in a single database, creating a generic model. On the contrary, we propose to predict stock price movements considering each stock as a distinct dataset, training specialized machine learning traditional classifiers for each one. We compare the proposed procedure, using different learners, mainly with state-of-the-art deep learning techniques. The results suggest that using specific models for each stock, employing simple and small feature sets, significantly contributes to improved model performance. Our best model, using Logistic Regression, outperformed all the other models.**

***Index Terms*—Stock Movement Forecast, Time Series, Machine Learning**

I. Introduction

Stock price predictions have attracted several investors who want to maximize their profits since the opportunities to expand their earnings are higher than investing in conventional financial approaches, such as savings or fixed deposits [1], [2]. The stock market is a complex system due to its nonlinear and non-stationary characteristics [3], [4]. This complexity is associated with factors such as political events, market news, quarterly earnings reports, international influence, and conflicted trading behavior [5].

Some statistical techniques for stock price prediction, such as ARMA, ARIMA, and GARCH, assume that a linear process can generate the time series [4], [6], [7]. However, these techniques do not produce an easily automated process as it necessitates adaptation and changes at each stage, which requires certain regularities and stationary nature in the target data. As a result, traditional statistical methods cannot be used to track the complexity and non-stationary nature of the stock market [4]. On the other hand, market analysts, traders, and researchers have investigated techniques such as Bayesian model, Fuzzy classifier, Artificial Neural Networks (ANN), Support Vector Machine (SVM) classifiers, and so on, to analyze stock markets and make trading decisions [4], [8].

We can categorize stock price forecasting techniques into technical analysis, fundamental analysis, and those that combine both. The first approach assumes that the future behavior of a financial time series is conditioned by its past. The second approach is based on economic and financial performance of a company and is affected by external factors, such as political and economic variables. The third approach considers relevant information from both technical and fundamental analysis [9].

According to the Efficient Market Hypothesis, fundamental analysis, technical analysis, or indeed any other strategy cannot generate above-average profits. As a result, the *buy-and-hold* strategy, where stocks are purchased and held until a decision to divest, would be considered the best strategy [10]–[12]. However, some studies contest this hypothesis. Haugen [13] analyzed the deficiencies of this hypothesis, and Los *et al.* [14] demonstrated that none of the six major Asian markets exhibit such behavior. Furthermore, practical experience in stock market trading indicates the existence of price trends and suggests that attempts to predict future trends can generate favorable returns [12].

Predicting the price trend of a specific stock means forecasting whether its price will rise or fall in the future: day, week, month, etc. In other words, considering t as the current day, the objective is to predict whether there will be an increase or decrease in its price on day $t + n$. This scenario can be seen as a binary classification problem, where the goal is to predict one of two classes, based on the input data.

In this work, we propose an approach for predicting stock price movements using traditional classifiers and treating each stock as a single dataset. More specifically, we tune hyper-parameters and train a specialized model for each stock. Our methodology follows a technical analysis approach that considers the percentage change of prices (closing, opening, high, low) and volume relative to their previous day values as features. Additionally, we incorporate the percentage change of these indicators related to their 5-day, 10-day, and 15-day moving averages. We claim that specialized models for each stock combined with feature extraction and data preprocessing play crucial roles in the classifier performance.

Our approach has been compared to other state-of-the-art proposals, outperforming them. The best result was obtained with the Logistic Regression classifier, showing that even traditional classifiers, not usually used for stock market prediction, can perform well given the appropriate attributes and specialized training for each stock.

The structure of our paper is as follows. Section II shows a brief literature review encompassing the most relevant works on trend forecasting in stock prices, introducing fundamental concepts that are essential for understanding our research. Section III gives a detailed explanation of the proposed method and Section IV presents the setup and results of the experiments run to evaluate it. Finally, Section V provides final remarks and potential possibilities for future research.

II. Background

Various techniques have been proposed to predict both the future price and the trend of stock prices: *up* or *down*. The performance of the proposed approaches varies significantly from one work to another. This fact can be explained by the size of the datasets, the models, and/or the techniques used for training and testing the data, e.g. cross-validation, walk-forward testing [12], etc. Some of these approaches are presented below.

Teixeira and Oliveira [12] use the *k-Nearest Neighbors* (kNN) classifier and 22 features derived from four technical indicators: Moving averages, Relative Strength Index (RSI), stochastics, and Bollinger bands to predict the trend in prices of 15 stocks from the Brazilian stock exchange B3, collected from 01/04/1998 to 09/03/2009.

The classifier presents three possible outputs: buy, sell, or hold. The authors consider that, when the price of a stock falls by more than 3% or rises by more than 10%, all shares in custody are sold. In addition, the RSI filter was also used to prevent any purchases when the index is over 70.

The full dataset was divided into ten subsets, each containing approximately one year of trades. A walk-forward sliding window strategy was used, sequentially selecting three subsets for training and the following subsets for testing. The main parameter used to measure the performance of the experiments was the profit earned during the test period. Performance was compared to the *buy-and-hold* approach. The results show their method performs better for 12 of the 15 stocks considered in their experiments.

Nguyen, Shirai, and Velcin [15] propose a model for predicting stock price movements based on historical stock prices and sentiment analysis in social networks. The model proposed by the authors incorporates specific sentiments on topics related to a company: data from 18 stocks obtained over one year were considered. The authors claim their results show that sentiment analysis can improve the performance of stock price forecasting models. They used the Support Vector Machine (SVM) classifier with linear *kernel* as the classification model and obtained a 54.41% accuracy on the proposed model, outperforming the compared approaches.

Nelson, Pereira, and Oliveira [16] investigated the use of deep learning to predict movements in stock prices, using a *Long Short-Term Memory* (LSTM) neural network model, and compare the results obtained with machine learning techniques. The authors tested five stocks from the Brazilian stock exchange, B3, from 2008 to 2015, considering historical values (open, close, high, low, volume) and 175 technical indicators, composing 180 attributes.

The model was trained and evaluated in a sliding window format, in which a new model is generated at the end of each trading day. In this approach, the preceding ten months leading up to the current day were utilized for training, while the model was validated using data from the most recent week. The test was conducted using December 2014 data, and the average outcome for each company was calculated. The authors achieved a 55.9% accuracy as their best result, specifically with the Bradesco bank share BBDC4.

Wu *et al.* [17] proposed a new deep learning model Cross-modal attention based Hybrid Recurrent Neural Network (CH-RNN). They used short texts from Twitter and historical stock prices to develop a predictive model for stock price movements. The authors created and shared a database containing data from January 2017 to November 2017, covering 231 days and including 47 stocks listed in the Standard & Poor's 500 index (S&P 500), having a significant number of available tweets. To conduct their experiments, the authors divided the data into training, validation, and testing sets, approximately following a chronological ratio of 5:1:1. They achieved a 59.15% accuracy rate with the proposed approach.

Liu *et al.* [18] proposed a Capsule network based on Transformer Encoder (CapTE) to extract deep semantic features from social media texts and used the same database utilized in [17]. The authors claim their approach outperformed the previous work, achieving an accuracy of 64.22% and a Matthews Correlation Coefficient (MCC) [19] of 0.3481. Furthermore, a profit analysis in both CapTE and CH-RNN showed that CapTE yielded the highest profit in five out of the six analyzed stocks.

The number and type of stocks, dataset size, split strategy, etc. can significantly impact the performance of the tested models. Consequently, comparing techniques that employ distinct methodologies is challenging. To address this issue, we compare our work with two state-of-the-art approaches already evaluated in the same training, validation, and testing sets [20]. The following sections provide details of these two approaches to enhance the comprehension of the shared common aspects utilized in our work.

A. Stocknet

Xu and Cohen [21] proposed StockNet, a generative deep-learning model for predicting stock price movements. The approach is based on Variational Auto-Encoders (VAE). A VAE consists of two independent models: an *encoder*, or recognition model, and a *decoder*, or generative model. The encoder takes input examples X from the database and converts them into a representation in an alternative dimension called *latent variables*, represented as Z. In the original approach, Z is used as input by the decoder, which converts it back to the original dimension X'. The closer X' is to X, the better the performance of the model.

The authors introduced a novel decoder that maps the latent variable Z to the price movement of a stock Y. The predictions are made for the target day d_t and other days in a lag of k trading days preceding the target day. The accuracy of the forecast Y is enhanced by combining the predictions for the target day with those for other days within the lag.

The authors collected and used a dataset comprising the prices (close, high, and low) of 88 stocks from NASDAQ and NYSE, collected between 01/01/2014 and 01/01/2016. This data was divided into three sets: training (01/01/2014 to 02/08/2015), validation (03/08/2015 to 30/09/2015), and testing (01/10/2015 to 01/01/2016). In addition to prices, they utilize Twitter messages related to the stocks.

Rather than using the actual price values, they focus on the variations relative to the previous day, normalized by the closing price. Considering p_t^s as the price of a stock s on day t, the percentage movement is given by $p_{t+1}^s/p_t^s - 1$. If this difference represents a value $\geq$ 0.055%, the label is set as 1 (increase), and if the value is $\leq$ -0.05%, it is set as 0 (decrease). Examples with values between -0.05% and 0.055% are removed: they account for 38.72% of the dataset. The thresholds 0.055% and -0.05% were used to balance the data of the classes.

The authors achieved a performance of 58.23% accuracy and an MCC of 0.080796 by combining stock prices with Twitter messages in their approach. When considering only the stock prices as input to the model, they obtained an accuracy of 54.96% and an MCC of 0.016456.

B. Adv-ALSTM

Feng *et al.* [20] assess that the methods commonly utilized for forecasting stock price movements achieve limited generalization. This limitation arises from the stochastic nature of stock prices, which is typically overlooked by these techniques. To address this issue, the authors propose employing adversarial training, which involves introducing small perturbations to the model to simulate the stochastic nature of stock prices. So, the authors propose Adv-ALSTM, which utilizes adversarial training in an ALSTM [22] network.

The authors conducted their experiments in two datasets: ACL18 [21] and KDD17 [23]. The KDD17 dataset consists of 50 stocks and covers nine years, from 01/01/2007 to 01/01/2016. The data was labeled using the same methodology employed in [21]. The data were divided into training (01/01/2007 to 01/01/2015), validation (02/01/2015 to 01/01/2016), and test (02/01/2016 to 01/01/2017) sets, following the same separation process described in [21] for the ACL18 database.

The authors evaluated their model, Adv-ALSTM, by comparing it with state-of-the-art techniques, including StockNet. They trained all the models using their 11 attributes. The experimental results demonstrated that Adv-ALSTM outperformed all the compared approaches. Specifically, on the ACL18 dataset, the model achieved 57.20% accuracy and 0.1483 MCC, while on the KDD17, it achieved 53.05% accuracy and 0.0523 MCC.

C. Time Series

A time series is a sequence of observations taken sequentially in time. Formally, a time series of length n can be represented by the notation shown in Equation 1 [24]. The equation comprises a set of n values sampled at discrete time points $1, 2, \ldots, n$. The notation can be shortened to x_t when the length n of the series is not explicitly mentioned [25].

$$x_t : t = 1, \ldots, n = x_1, x_2, \ldots, x_n \tag{1}$$

A fundamental concept in time series analysis is *stationarity*. For a time series x_t to be considered strictly stationary, it must exhibit statistical equilibrium, meaning that its statistical properties remain unchanged over time t. In other words, a series is strictly stationary if the joint probability distribution of $x_{t_1}, ..., x_{t_n}$ is the same as the joint distribution of $x_{t_{1+m}}, ..., x_{t_{n+m}}$ for all values of $t_1, ..., t_n$ and any integer quantity m [25], [26].

Strict stationarity implies that the mean and variance of a time series remain constant over time. When two examples in a time series, represented as x_t and x_s, are correlated, and this correlation depends solely on the number of time intervals that separate them, we call the series a second-order stationary series. Specifically, considering variables x_t and x_s, the correlation between them is determined by the interval $k = |t - s|$. The term *lag* is used to refer to the number of time intervals k between the variables [25], [26].

Non-stationary series can be transformed into stationary ones through a procedure called differencing, which involves subtracting adjacent terms of the series, as shown in Equation 2 [25], [27].

$$\nabla_{x_t} = x_t - x_{t-1} \tag{2}$$

In the context of time series analysis, a systematic change in the series that does not follow a periodic pattern is referred to as a *trend*. The simplest form of a trend is linear growth or decline. On the other hand, when a pattern repeats within a fixed interval, such as the increased number of reservations at a restaurant on specific days of the week, it is termed *seasonality* [25]. A time series with a trend or with seasonality is considered non-stationary [28].

Stationarity is an idealization that pertains to models. When fitting a stationary model, it is assumed that the data represents a realization of a stationary process. Consequently, the initial step in time series analysis should involve checking for any trend or seasonality and eliminating them [25].

Livieris *et al.* [29] claim that transforming a non-stationary into a stationary series is expected to improve forecasting performance, when compared to the same model trained on the original series. In our work, we used differentiation on all datasets. The stationarity of the differentiated series was evaluated based on the Augmented Dickey-Fuller test [30]. In our case, only one differentiation was sufficient to make our database series stationary.

III. Proposed Method

This paper proposes an approach to predicting the price trend of a stock, i.e., whether the price will rise or fall the next day. For this purpose, we create one optimized classification model for each share, based on traditional machine learning classifiers. As features, we use the percentage change of an indicator (opening price, closing price, highest price, lowest price, and volume) concerning its value on the previous day. In addition to these attributes, we investigated the percentage variations of the same indicators relative to their respective moving averages. Our technique encompasses three distinct components: *feature extraction*, *optimization*, and *testing*.

A. Feature Extraction

During the feature extraction process, the original prices (open, close, high, low) and volume were replaced by their corresponding percentage changes, related to their previous day values. The variations, related to their respective m moving averages, are also considered. The original values were adjusted to consider corporate events such as splits and reverse splits in order to ensure data consistency and proportionality.

The percentage change (pc) of observation x_t in relation to observation x_{t-1} is calculated using Equation 3. The numerator represents the differentiation of the series. The differentiation is normalized by x_{t-1} to indicate the variation of x_t concerning x_{t-1}. For example, a 0.1 change in the closing price of a stock signifies a 10% variation from the previous day.

$$pc(t) = \frac{x_t - x_{t-1}}{x_{t-1}} = \frac{\nabla_{x_t}}{x_{t-1}} \tag{3}$$

Equation 3 was applied to each original price and volume considered in this work, resulting in five new attributes, pc_o, pc_c, pc_h, pc_l and pc_v, corresponding to the percentage variation of the attributes opening price, closing price, highest price, lowest price, and volume, respectively. The original values were discarded after feature extraction.

In addition to computing the percentage variation of the attributes, we also calculated the percentage variation of each indicator relative to specific moving averages. Our model was designed to forecast the upcoming trading day. Thus, we decided to incorporate the 5, 10, and 15-day moving averages for each value. These nearest moving averages offer valuable insights into shorter-term trends. The k-th day moving average is calculated by Equation 4.

$$ma_k(t) = \frac{1}{k} \sum_{i=t-k+1}^{t} x_i \tag{4}$$

where t is the index of the current day and $ma_k(t)$ is the moving average of k days on day t.

The percentage change of the original (price or volume) value to the k-day moving average (pcm) is calculated using equation 5. Note that $pcm_k(t)$ denotes the difference between the original value and the k-day moving average on the day t.

$$pcm_k(t) = \frac{x_t - ma_k(t)}{ma_k(t)} \tag{5}$$

It is important to observe that the procedure of calculating the percentage variation of the indicators relative to the moving averages derives a set of 15 additional attributes. Combined with the five attributes obtained from Equation 3, they form the 20 features shown in Table I. Note that the suffixes *o, c, h, l* and *v* at each attribute name are used to represent open, close, high, low, and volume, respectively.

TABLE I
ATTRIBUTES USED IN THIS WORK

Indicator	pc	pcm$_5$	pcm$_{10}$	pcm$_{15}$
Opening Price	pc_o	pcm_5o	$pcm_{10}o$	$pcm_{15}o$
Closing Price	pc_c	pcm_5c	$pcm_{10}c$	$pcm_{15}c$
Highest Price	pc_h	pcm_5h	$pcm_{10}h$	$pcm_{15}h$
Lowest Price	pc_l	pcm_5l	$pcm_{10}l$	$pcm_{15}l$
Volume	pc_v	pcm_5v	$pcm_{10}v$	$pcm_{15}v$

B. Optimization

Unlike the approaches used by Xu *et al.* [21] and Feng *et al.* [20], this work considers the unique characteristics of each share by treating them as individual datasets. As a result, we have 50 datasets (see Table III). After the feature extraction, the classification models are optimized using the validation set, resulting in a specialized model for each dataset. So the predictions for each stock are made by a dedicated model tailored to its specific characteristics rather than relying on a generic model.

K-Nearest Neighbors (kNN), Naive Bayes (NB), Support Vector Machines (SVM), and Logistic Regression (LR) classifiers were utilized in this study as they are some of the most popular classifiers in the machine learning area. The grid search algorithm [31] utilized the validation set to identify the optimal hyperparameters for each dataset. Table II shows the hyperparameters used by grid search. The best hyperparameters yielded by the algorithm were used to train the model and evaluate its performance on the test set.

TABLE II
CLASSIFIERS AND HYPERPARAMETERS INPUT SET

Classifier	Hyperparameters
LR	C: [1, 2, 3, 4, 5], tol: [0.1, 0.2, 0.3, 0.4, 0.5], solver: ['lbfgs', 'liblinear', 'newton-cg', 'sag', 'saga']
KNN	n_neighbors: [5, 10, 15, 20, ..., 45]
SVM	C: [0.01, 0.1, 1, 10, 100], kernel: ['poly', 'linear']
NB	—

C. Testing

The best model obtained through the grid search algorithm for each dataset is applied to the corresponding test set to evaluate the classifier performance on each stock. This process is performed for all datasets. The overall model performance is calculated by averaging the results in all sets.

IV. Experiments

This section describes the experiments setup and results. They were run in Python, using the Scikit-learn library [32]. To evaluate the model's performances, we used the metrics: accuracy, AUC, MCC, precision, recall, and F1 score.

The experiments were conducted using the KDD17 dataset [23], as employed by Feng *et al.* [20]. This dataset is publicly available on GitHub[1] and consists of 50 stocks from the US markets. These stocks are categorized into ten sectors, as illustrated in Table III. From each sector, the five highest market capitalization stocks were selected. The dataset covers the period from 01/01/2007 to 01/01/2016 and was divided into training (01/01/2007 to 01/01/2015), validation (02/01/2015 to 01/01/2016), and test (02/01/2016 to 01/01/2017) sets.

Following the approach adopted by Feng *et al.* [20] and Xu *et al.* [21], instances with a close price movement percentage greater than or equal to 0.55% and less than or equal to -0.5% are classified as positive (up) and negative (down), respectively, as outlined in Section II-A. Examples outside this range were removed from the database to balance the classes.

TABLE III
STOCK SYMBOLS AND SECTORS OF THE SELECTED CORPORATIONS

Basic Materials	Cyclicals	Energy	Financials	Healthcare
BHP	AMZN	CVX	BAC	JNJ
BA	KO	AAPL	CHL	D
DOW	CMCSA	PTR	BRK-B	MRK
GE	MO	GOOGL	DCM	DUK
RIO	DIS	RDS-B	JPM	NVS
Industrials	**Non-Cyclicals**	**Technology**	**Telecommunications**	**Utilities**
MA	PEP	INTC	NTT	EXC
SYT	HD	TOT	SPY	PFE
MMM	PG	MSFT	T	NGG
VALE	TM	XOM	WFC	UNH
UPS	WMT	ORCL	VZ	SO

A. Results

Regarding the results, Table IV shows the performance of the models utilizing our approach, including all the selected metrics, whereas Table V compares our results to those of the state-of-the-art techniques, with a specific focus on Accuracy and MCC, the only metrics used by the compared models. For simplicity, we considered the results of Feng *et al.* [20] for the compared techniques. All the results are shown using the mean and standard deviation in the 50 databases considered and the best results are written in bold.

TABLE IV
MODEL PERFORMANCES: MEAN ± STANDARD DEVIATION

Measure	KNN	NB	SVM	LR
Accuracy	0.5110±0.0493	0.5287±0.0341	0.5317±0.0434	**0.5356**±0.0456
AUC	0.5104±0.0487	0.5201±0.0290	0.5102±0.0218	**0.5224**±0.0407
MCC	0.0211±0.0985	0.0443±0.0661	0.0372±0.0688	**0.0578**±0.0980
Precision	0.5486±0.0566	0.5602±0.0417	**0.5865**±0.1440	0.5637±0.0653
Recall	0.5181±0.1098	0.6211±0.2144	**0.7379**±0.3665	0.6542±0.2555
F1	0.5272±0.0729	0.5653±0.1211	0.5626±0.2416	**0.5752**±0.1368

In the results of Table IV, using our proposal, Logistic regression obtained the best results, outperforming all other models in accuracy, AUC, MCC, and F1. It also obtained the second-best performance in Precision and Recall: 0.5637 and 0.6542, respectively. The SVM model performed the best in Precision and Recall, achieving values of 0.5865 and 0.7379, respectively. The Naive Bayes model yielded similar results to the SVM models, while the KNN model exhibited the lowest performance among the models utilizing our approach.

[1] https://github.com/hennande/Adv-ALSTM

TABLE V
COMPARISON OF EVALUATED MODELS

Model	Accuracy	MCC
MOM	49.75±—	-0.0129±—
MR	48.46±—	-0.0366±—
StockNet	51.93±5e-3	0.0335±5e-3
LSTM	51.62±4e-1	0.0183±6e-3
ALSTM	51.94±7e-1	0.0261±1e-2
Adv-ALSTM	**53.05**±—	**0.0523**±—
Using our approach		
KNN	0.5110±0.0493	0.0211±0.0985
Naive Bayes	0.5287±0.0341	0.0443±0.0661
SVM	0.5317±0.0434	0.0372±0.0688
Logistic Regression	**0.5356**±0.0456	**0.0578**±0.0980

Even though most of the techniques in Table V use deep learning methods, our best model (Logistic Regression) delivered superior performance, surpassing all the compared state-of-the-art models in accuracy and MCC.

Note that the SVM classifier also outperformed all the state-of-the-art models in accuracy. However, in MCC, the Adv-ALSTM model has a better result. Also, the Naive Bayes classifier outperformed the state-of-the-art models in both accuracy and MCC, except for Adv-ALSTM. Our worst performance was observed with the KNN classifier, surpassing only two models: MOM and MR.

To conclude, it is worth adding that we have also tried to use concept drift detectors [33] such as RDDM [34] combined with both single classifiers and some ensembles [35] but the results did not improve.

V. CONCLUSIONS

This paper proposed an approach for predicting stock price movements using technical indicators based on the percentage change of prices, volume, and related moving averages. It considers each stock as a distinct dataset and trains a specialized classifier for each one. We compare the proposed procedure mainly with state-of-the-art deep learning techniques.

The results demonstrate that a specialized model per stock, employing simple and small feature sets, can outperform generalized state-of-the-art deep learning models in both accuracy and MCC. When training a model, it is essential to consider the specific characteristics of each stock: combining all shares and training a single classifier may ignore certain specificities. We argue that the model can effectively capture these particularities using specialized training. The results suggest that a comprehensive understanding of the data, feature extraction, and the use of specific models for each stock significantly contribute to improve the model performance. Even simpler models can achieve good performance.

As future work, we propose to explore a hybrid approach that leverages an ensemble of classifiers. Such approach could combine specialized models for individual stocks with generic models trained on all stock data.

ACKNOWLEDGMENT

José Junior O. Silva is on leave from Instituto Federal de Alagoas (IFAL). Roberto S. M. Barros is supported by research grant number 305539/2022-1 from Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). Silas Santos was previously supported by post-doctorate grant number 88887.374884/2019-00 from Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES).

DISCLAIMER

The views and opinions expressed in this research are solely those of the authors and do not necessarily reflect the official policy or position of the company.

REFERENCES

[1] A. Thakkar and K. Chaudhari, "A comprehensive survey on portfolio optimization, stock price and trend prediction using particle swarm optimization," *Archives of Computational Methods in Engineering*, vol. 28, no. 4, pp. 2133–2164, Jun. 2021. [Online]. Available: https://link.springer.com/10.1007/s11831-020-09448-8

[2] A. S. I. Pramanaswari and G. W. Yasa, "Graham & Dodd theory in stock portfolio performance in LQ 45 index at Indonesia stock exchange," *International research journal of management, IT and social sciences*, vol. 5, no. 6, pp. 52–59, Nov. 2018, number: 6. [Online]. Available: https://sloap.org/journals/index.php/irjmis/article/view/338

[3] R. Nazário, J. Silva, V. Sobreiro, and H. Kimura, "A literature review of technical analysis on stock markets," *The Quarterly Review of Economics and Finance*, vol. 66, pp. 115–126, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1062976917300443

[4] R. Bisoi and P. Dash, "A hybrid evolutionary dynamic neural network for stock market trend analysis and prediction using unscented Kalman filter," *Applied Soft Computing*, vol. 19, pp. 41–56, Jun. 2014.

[5] J. L. Ticknor, "A Bayesian regularized artificial neural network for stock market forecasting," *Expert Systems with Applications*, vol. 40, no. 14, pp. 5501–5506, Oct. 2013.

[6] H. Liu and B. Song, "Stock price trend prediction model based on deep residual network and stock price graph," in *11th International Symposium on Computational Intelligence and Design (ISCID)*, vol. 02, Dec. 2018, pp. 328–331.

[7] D. Kumar and S. Murugan, "Performance analysis of Indian stock market index using neural network time series model," in *Intern. Conf. on Pattern Recognition, Informatics and Mobile Eng.*, 2013, pp. 72–78.

[8] D. P. Gandhmal and K. Kumar, "Systematic analysis and review of stock market prediction techniques," *Computer Science Review*, vol. 34, p. 100190, Nov. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S157401371930084X

[9] M. R. Vargas, B. Lima, and A. Evsukoff, "Deep learning for stock market prediction from financial news articles," in *IEEE Intern. Conf. on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, Jun. 2017, pp. 60–65.

[10] E. F. Fama, "The behavior of stock-market prices," *The Journal of Business*, vol. 38, no. 1, pp. 34–105, 1965, University of Chicago Press. [Online]. Available: https://www.jstor.org/stable/2350752

[11] W. Jiang, "Applications of deep learning in stock market prediction: Recent progress," *Expert Systems with Applications*, vol. 184, p. 115537, Dec. 2021.

[12] L. Teixeira and A. L. I. Oliveira, "A method for automatic stock trading combining technical analysis and nearest neighbor classification," *Expert Systems with Applications*, vol. 37, no. 10, pp. 6885–6890, 2010.

[13] R. A. Haugen, *The New Finance: The Case Against Efficient Markets*, 1st ed. Englewood Cliffs, N.J.: Pearson, Nov. 1994.

[14] C. A. Los, "Nonparametric efficiency testing of asian stock markets using weekly data," *Finance*, Sep. 2004, number: 0409033, University Library of Munich, Germany. [Online]. Available: https://ideas.repec.org//p/wpa/wuwpfi/0409033.html

[15] T. H. Nguyen, K. Shirai, and J. Velcin, "Sentiment analysis on social media for stock movement prediction," *Expert Systems with Applications*, vol. 42, no. 24, pp. 9603–9611, Dec. 2015.

[16] D. M. Nelson, A. C. M. Pereira, and R. A. Oliveira, "Stock market's price movement prediction with LSTM neural networks," in *Intern. Joint Conf. on Neural Networks (IJCNN)*, May 2017, pp. 1419–1426.

[17] H. Wu, W. Zhang, W. Shen, and J. Wang, "Hybrid deep sequential modeling for social text-driven stock prediction," in *Proceedings of 27th ACM International Conference on Information and Knowledge Management (CIKM'18)*, New York, NY, USA, 2018, pp. 1627–1630.

[18] J. Liu, H. Lin, X. Liu, B. Xu, Y. Ren, Y. Diao, and L. Yang, "Transformer-based capsule network for stock movement prediction," in *Proceedings of the First Workshop on Financial Technology and Natural Language Processing*, Macao, China, Aug. 2019, pp. 66–73. [Online]. Available: https://aclanthology.org/W19-5511

[19] B. W. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochimica et Biophysica Acta (BBA) - Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.

[20] F. Feng, H. Chen, X. He, J. Ding, M. Sun, and T.-S. Chua, "Enhancing Stock Movement Prediction with Adversarial Training," in *Proceedings of 28th International Joint Conference on Artificial Intelligence, IJCAI-19*, Jul. 2019, pp. 5843–5849. [Online]. Available: https://doi.org/10.24963/ijcai.2019/810

[21] Y. Xu and S. B. Cohen, "Stock Movement Prediction from Tweets and Historical Prices," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia, Jul. 2018, pp. 1970–1979. [Online]. Available: https://aclanthology.org/P18-1183

[22] Q. Wang and Y. Hao, "ALSTM: An attention-based long short-term memory framework for knowledge base reasoning," *Neurocomputing*, vol. 399, pp. 342–351, Jul. 2020.

[23] L. Zhang, C. Aggarwal, and G.-J. Qi, "Stock Price Prediction via Discovering Multi-Frequency Trading Patterns," in *Proc. of 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '17, New York, NY, USA, 2017, pp. 2141–2149. [Online]. Available: https://dl.acm.org/doi/10.1145/3097983.3098117

[24] G. Box, G. Jenkins, G. Reinsel, and G. Ljung, *Time Series Analysis: Forecasting and Control*, ser. Wiley Series in Probability and Statistics. Wiley, 2015. [Online]. Available: https://books.google.com.br/books?id=rNt5CgAAQBAJ

[25] A. V. Metcalfe and P. S. Cowpertwait, *Introductory Time Series with R*, ser. Use R! Springer New York, 2009. [Online]. Available: https://doi.org/10.1007/978-0-387-88698-5

[26] J. D. Hamilton, *Time series analysis*. Princeton, N.J: Princeton University Press, 1994.

[27] V. Kotu and B. Deshpande, "Chapter 12 - time series forecasting," in *Data Science*, 2nd ed. Morgan Kaufmann, 2019, pp. 395–445.

[28] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 3rd ed. Melbourne, Australia: OTexts, 2021. [Online]. Available: OTexts.com/fpp3

[29] I. E. Livieris, S. Stavroyiannis, E. Pintelas, and P. Pintelas, "A novel validation framework to enhance deep learning models in time-series forecasting," *Neural Computing and Applications*, vol. 32, no. 23, pp. 17 149–17 167, Dec. 2020.

[30] Y.-W. Cheung and K. S. Lai, "Lag order and critical values of the augmented Dickey–Fuller test," *Journal of Business & Economic Statistics*, vol. 13, no. 3, pp. 277–280, 1995, Taylor&Francis.

[31] P. Liashchynskyi and P. Liashchynskyi, "Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS," Dec. 2019. [Online]. Available: https://arxiv.org/abs/1912.06059v1

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[33] R. S. M. Barros and S. G. T. C. Santos, "A large-scale comparison of concept drift detectors," *Information Sciences*, vol. 451-452, pp. 348 – 370, 2018.

[34] R. S. M. Barros, D. R. L. Cabral, P. M. Gonçalves Jr., and S. G. T. C. Santos, "RDDM: Reactive drift detection method," *Expert Systems with Applications*, vol. 90, no. C, pp. 344–355, 2017.

[35] R. S. M. Barros and S. G. T. C. Santos, "An overview and comprehensive comparison of ensembles for concept drift," *Information Fusion*, vol. 52, no. C, pp. 213–244, 2019.