TRX Cryptocurrency Profit and Transaction Success Rate Prediction Using Whale Optimization-Based Ensemble Learning Framework

Amogh Shukla 1, Tapan Kumar Das 2 and Sanjiban Sekhar Roy 1,*

1 School of Computer Science and Engineering, Vellore Institute of Technology, Vellore 632014, Tamil Nadu, India
2 School of Information Technology and Engineering, Vellore Institute of Technology, Vellore 632014, Tamil Nadu, India
* Correspondence: s.roy@vit.ac.in

Abstract: TRON is a decentralized digital platform that provides a reliable way to transact in cryptocurrencies within a decentralized ecosystem. Thanks to its success, TRON’s native token, TRX, has been widely adopted by a large audience. To facilitate easy management of digital assets with TRON Wallet, users can securely store and manage their digital assets with ease. Our goal is first to develop a methodology to predict the future price using regression and then move on to build an effective classifier to predict whether a profit or loss is made the next day and then make a prediction of the transaction success rate. Our framework is capable of predicting whether there will be a profit in the future based on price prediction and forecasting results using regressors such as XGBoost, LightGBM, and CatBoost with $R^2$ values of 0.9820, 0.9825 and 0.9858, respectively. In this work, an ensemble-based stacking classifier with the Whale optimization approach has been proposed which achieves the highest accuracy of 89.05 percent to predict if there will be a profit or loss the next day and an accuracy of 98.88 percent of TRX transaction success rate prediction which is higher than accuracies obtained by standard machine learning models. An effective framework will be useful for better decision-making and management of risks in a cryptocurrency.

Keywords: blockchain; cryptocurrency; Tron; machine learning; profit prediction; TRX token; ensemble learning

MSC: 68T05

1. Introduction

Blockchain technology has emerged as a game-changer in the wake of the success of Bitcoin and other cryptocurrencies. Decentralized application developers tend to instead focus on finding a solution that best meets their needs and the needs of their decentralized applications market rather than developing a new blockchain platform from scratch as the number of existing ones grows. Decentralization-based technologies guarantees the ownership and transferability of digital financial assets to a certain extent. The upsurge in market value and popularity of cryptocurrencies raises several global issues and concerns for business and industrial economics. Distributed ledger has gained traction over the years and refers to the technological infrastructure and protocols to meet the very objectives of transparency and traceability. For each secure transaction, a separate secure location has to be maintained over the internet by the participants, who are essentially the users of the distributed ledger technology. Blockchain makes it possible to maintain a widely dispersed network through cooperative efforts to establish mechanisms for coordination, communication, and motivation [1]. The term “blockchain finance” refers to how this technology has been implemented. This has given rise to several digital platforms, such
as TRON, which has its own digital coin, called the TRX coin. The usage of blockchain technology in the finance sector reduces costs and improves service quality in a safe and flexible manner [2]. The Blockchain distributed database is nearly impossible to alter, whereas traditional credit models rely on third parties to process payment information. As a result of peer-to-peer innovation and decentralization in the financial sector, there has been rapid global expansion and a trustworthy database.

The basis of a blockchain is an immutable, tamper-proof chain of linked blocks. Sets are recorded in each block, and the metadata are associated with a transaction. It was launched as a peer-to-peer currency exchange system. Blockchain topologies act as a critical parameter in developing a blockchain application by classifying the nodes that will be a part of it. Permissioned, private, and hybrid blockchain systems are the most recent subdivisions of blockchain technology [3,4]. Transactions on the Blockchain have the same effect on both parties; each node has its ledger [5]. The role of digital currency tokens, often referred to as crypto-tokens, have a unique role in transforming the way people make transactions over the web. This has become more useful since the introduction of the first cryptocurrencies such as Bitcoin [6] and Ethereum [7] that have nodes that can generate the next valid block without consensus [8]. TRX serves a similar function fundamentally, to make safe and secure transactions digitally. The TRON Protocol, one of the biggest operating systems based on blockchain, provides scalability, availability, and high throughput for the Decentralized Applications (Daps) in the TRON environment [9,10]. TRON Protobuf is a serialization mechanism for structured data, similar to JSON or XML but significantly faster and smaller [11]. A custom-built blockchain network, such as TRON, may be more efficient, convenient, secure, and stable for millions of global developers [12]. For now, it is impossible to predict how high TRON (TRX Token) will rise. The primary goal of the TRON network is to decentralize the distribution and content creation industry, which has been criticized for censorship and revenue distribution unfairly. Without a doubt, as individuals, it is our own responsibility to act ethically when it comes to digital media and always take responsibility for the content we post online or on any digital media platform. Additionally, TRON provides a few other tools to help ensure more democratic content creation. TRON was reasonably conceived as an innovative solution to address robust scalability issues and appears to be like its competitors on the surface. All of them could just as easily create Daps that further motivate the increase in the adoption of TRON. Quite a few of them already have. That being said, TRON still has the potential to succeed better in this market if its adoption increases over time. It is possible that the growing importance of interoperability in the blockchain industry will help TRON maintain its position as one of the leading networks for the distribution of original digital content. TRON’s unique selling point is that anyone, anywhere, with an internet connection, can access data on the network. Several researchers have made useful attempts in the blockchain and cryptocurrency sphere leading up to the present, where these works validate the usefulness of various Artificial Intelligence techniques in the technology. Very few attempts, if any, have been made to study TRON as a cryptocurrency by applying Artificial Intelligence to predict profit the next day, prices in the future, and address transaction success and failure rates. The scope and flexibility of the research and a framework that can be applied when analyzing digital tokens, as well as other cryptocurrencies having a relatively good standing in the market.

The followings are highlights of the contributions of the paper:

- We checked the potential capabilities of regression models such as Catboost, Xgboost and Light GBM for predicting the future price of cryptocurrencies.
- We have proposed classification models for TRX transaction success rate powered by machine learning techniques.
- Finally, we have estimated the earnings from the profit made by the TRX tokens.

The rest of this paper is organized as follows: Section 2 is about the literature review pertaining to blockchain technology and cryptocurrency price prediction. Section 3 is about the dataset, visualizations and discussion about the detailed methodology involved in this
2. Literature Survey

The market value of the TRX coin is closely tied to the broader applications of blockchain technology, and to make accurate price predictions for specific coins, it is crucial to study related research findings and recent approaches. Factors such as market sentiment, regulatory changes, and technological advancements can greatly affect the value of TRX. It is also well argued that the price of a particular decentralized currency, just like TRX, follow other similar cryptocurrency on market trends. The reasons may be that it is seen as an asset, just like equity in the public market shareholdings (stock market), and is heavily dependent on market sentiments and the technology behind it. Therefore, we discuss these separately along with the literature on cryptocurrency price predictions.

2.1. Literature on the Scope of Cryptocurrencies and the Blockchain Technology

In this section, we survey several papers related to our work and group them accordingly. The key findings, mechanisms, and objectives are discussed. Vergne et al. [13], provide insights on Blockchain, machine learning (ML), and the future of digital platforms: centralized vs. distributed organizations. Sabry et al. [14] provide an overview of how artificial intelligence (AI) is being used to to find solutions for problems associated with cryptocurrency, including the inability of humans to keep up with the sheer volume of daily transactions, trades, and news. Recent studies in this field are discussed, with comparisons between them regarding methodology and data sets. It also calls attention to several improvement opportunities and research gaps. On the same lines, Bhutta et al. [15] explored the history, design, and safety of blockchain technology. Tanwar et al. [16] reviewed interdependent price prediction schemes for cryptocurrencies using deep learning techniques and predicted the value of cryptocurrencies using deep learning and their dependent relationships. They propose a hybrid model based on a deep learning approach to estimate the price of Litecoin and Zcash and the suggested model has been successfully trained and tested on industry-standard data sets, allowing immediate use in real-world settings. These cryptocurrencies are similar to TRX in the context of their relative market sizes. Štastný et al. [17], looked at dynamic time warping and machine learning, they create a model and cluster the top 30 cryptocurrency prices. Their elaboration on how AI is influencing cryptocurrency is useful to understand the essentials of price predictions. Godsiff et al. [18], in their work, ‘Bitcoin: Bubble or Blockchain’, where agent and multi-agent systems are discussed; applications and new technologies and provided a brief, comprehensive overview of the cryptocurrency that is managed by a distributed multi-agent system (DAS). Along with this, Bachani et al. [19] discuss different blockchain platforms that use the Delegated proof of stake (DPoS) consensus algorithm, including Tron, and explain the specific features and properties of each platform, such as the number of validators (or delegates), block creation time, block reward, and how validators are chosen. It is, therefore, safe to say that, given the traction, the cryptocurrencies have gained, Blockchain will see increased adoption as a technology and have a very strong technological background. TRON therefore, relies on some of the most innovative technologies ever developed and the TRX token has great potential in this regard.

2.2. Literature on Related Applications of Blockchain Technology

Alrowaily et al. [20] presented a study on a Peer-to-Peer Electronic cash systems for cryptocurrencies, including TRON. This study essentially suggests that it is overall easier to send electronic cash from one person to another without the involvement of a bank or other financial institutions using digital tokens. Yli-Huumo et al. [21] surveyed relevant scientific literature on Blockchain technology, and performed thorough mapping. They discuss the present research topics, future directions and challenges in Blockchain technology from a technological perspective. They, therefore, research the technology as
a whole, which suggests TRON environment ensures better transaction security with the consensus mechanism and its adoption will lead to a better ecosystem with the adoption of TRON wallet by its users. Mayer et al. [22] researched practical and interdisciplinary perspectives on distributed ledger technology. Mengelkamp E et al. [23] studied blockchain based smart grids. The increasing use of renewable energy necessitates a new pricing and distribution model for a volatile and decentralized generation. It is interesting how well these digital platforms are helping the physical environment which makes a safe and secure digital environment. Xu et al. [24] discussed blockchain immunity to some malicious attacks. The distributed consensus made it near-impossible to alter or compromise the public ledger, which records cryptographic transactions. Bengtsson et al. [25] investigated the issue and empirically quantified its effects. For each of these protocols, the potential stake distribution lag is several days, which they discovered after surveying existing secure PoS proposals. Yang et al. [26] examine the diffusion of risk in the cryptocurrency market from 2018 to 2021 and compare it to the stock and foreign exchange markets, finding that the risk diffuses more easily in the cryptocurrency market and is most affected by cryptocurrency with large market capitalizations or low-turnover. George et al. [27] performed a study on the distributed operating system (DOS) for permissioned blockchains. The first truly extensible blockchain system, Fabric, is used to run distributed applications. All of these papers discuss how effective blockchain has a future scope of adoption by the masses. Sayeed et al. [28] stated that Smart contracts were programmed and stored on decentralized blockchains and are activated when triggered. Earlier, Macrinici et al. [29] studied the applications of smart contracts within blockchain technology. It is an overview of current research trends in the field and a study of 64 papers organized according to the most important publication sources, channels, approaches, and methods discovered through their systematic mapping of research.

This literature on applications tends to focus on the future of these cryptocurrencies and thus the TRON as a platform, with its cryptocurrency, relies on how well structured the entire backbone of the blockchain technology is. A way to test this in practice is by increased adoption and utility of the Tron platform, which can, in turn, drive up demand for TRX as more users will need to interact with smart contracts and other services on the platform. Additionally, smart contracts can enable new use cases for TRX, such as decentralized exchanges and prediction markets, which contributes to increased demand and value. The efficiency and security of the consensus mechanism of a cryptocurrency can impact its adoption and attractiveness to users, which can in turn affect its value.

2.3. Literature on Cryptocurrency Price Prediction and Its Relevance with the TRX Token

Wu et al. [12] presented an overarching look into the network analysis of bitcoin transactions. They discuss the history of bitcoin transaction network analysis and a review of previous work in these areas, as well as in the related ones of network modeling, network profiling, and network-based detection, to give a unified framework for future study. Earlier a study by Valdivia et al. [30] focused on cryptocurrencies and the fallacy of decentralization. For two key reasons—privacy and independence from a central authority, cryptocurrencies have the potential to alter the financial services industry radically. This should be evident with network analysis of platforms such as TRON. Motamed et al. [31] studied the cryptocurrency transaction graphs quantitatively. They focused on the fact that there has not been a thorough comparison of other cryptocurrencies until then; instead, there have only been a few restricted statistical analyses of cryptocurrency transactions, most of which focused on Bitcoin. Later on, Crowcroft et al. [32] provided analysis of bitcoin price using user network data and verified transactions and in addition to standard features, they take into account the innovative concept of Trustful Transaction Graph, which was found to be useful for all the relevant parties and users in the blockchain networks. Gerritsen et al. [33] raised that bitcoin investors may make money off of the forecasts of cryptocurrency experts. They demonstrate that neutral and bearish forecasts for bitcoin published by crypto experts are followed by negative abnormal returns. However,
using a dataset they manually gathered, bullish expectations were not connected with any nonzero abnormal returns. Prediction updates are calculated using all active forecasts and comparing them to (i) the most recent forecast and (ii) the active consensus forecast. The transactions are sent to the servers by the users of the blockchain system. This suggests that bullish expectations do not have a significant impact on TRON’s abnormal returns but a study on the transaction success and failure rates and the next day’s profits, and a model to predict them, may solve the problem of manually analysing a large amount of data on these currencies and indirectly gives the user a way to judge how effective the transaction may be based on historical data [34]. Sebastião et al. [35] presented research focusing on using machine learning for cryptocurrency price prediction and trading in volatile markets. Characteristics from trading and network activity from 15 August 2015, to 3 March 2019, with the test sample starting on 13 April 2018, were used in the classification and regression procedures. Five of the eighteen individual models had success rates lower than 50% over the testing period. This is an important insight for developing accurate binary classification models for price prediction.

2.4. Key Research Findings and Recent Approaches for Cryptocurrency Price Analysis

Optimization is the process of finding the best solution or set of solutions to a problem, often by minimizing or maximizing a function or objective. Many optimization techniques have been developed to solve complex problems, including gradient-based methods [36], evolutionary algorithms [37], and swarm intelligence techniques [38]. Each of these methods has its strengths and weaknesses, making them more or less suitable for different types of problems [39]. The advantages of using whale optimization based ensemble learning are improved search capabilities, Enhanced diversity and Better performance [40] while drawbacks of previous optimization techniques might include Susceptibility to local optima [41], Convergence speed [42], Complexity and Parameter tuning [43] among others. We found two distinct approaches for price prediction that have recently been used widely and how these approaches are used for different use cases. These are (i) Attention-based ANNs where time series data are used for price prediction and various time series features are used to find correlation in the given context [44] and (ii) where previous market analysis is made, and the region of uncertainty is found, followed by prediction of profit. It is important to discuss both approaches to understand the two use cases. They both may use an open, high, low, and close price (OHLC) based data and correlate different features to make a statement on their predictions and analysis. Like in the case of price prediction by Song et al. in [44], a hybrid model utilizes a mechanism such as self-attention to extract the sequences in time series sequence links. The decoder then extracts the association between the values of the expected remaining time until a certain price level is reached. The input is a vector data $X = x_1, x_2, x_3, \ldots, x_n$ and returns a $q$ vector with indices $Z = 1, 2, \ldots, N$. The $d$-dimensional input is used with value arrays given by $(k_1, v_1), (k_2, v_2), (k_3, v_3), \ldots, (k_n, v_n)$. The mechanism is shown in Equation (1).

$$\text{attn}(K, V, q) = \sum_{i=1}^{N} \text{softmax}\left(\frac{k_i \cdot q}{\sqrt{d}}\right) \cdot v_i$$  \hspace{1cm} (1)$$

We use this kind of approach to use more relevant data in order to obtain different accuracy scores in a specific range. The function $f_w : X \rightarrow Y$ is used as a self-stepping function for prediction with $\Phi(v_i, \lambda)$, where the age is given by $\lambda$ [45]. With hyperparameter tuning, as the number of estimators is increased, the model reaches a better accuracy score, as shown in the training of their metrics.

$$\min_{i=1}^{N} \{v_i, \text{Loss}_{\text{Train}}(f_w(x_i), y_i) + \phi(v_i, \lambda)\}$$  \hspace{1cm} (2)$$

$$Y_n = \text{Periodic}_n + \text{Trend}_n + \text{Residual}_n$$  \hspace{1cm} (3)$$
Equation (3) shows the components of time-series decomposition. The study is widely used in time series price prediction, however, due to the nature of our dataset and the need for a different type of framework to be adapted, we focus on the approach (ii) and predict the future prices using regression and classification-based approaches.

3. Materials and Methods

A lot of money goes into it from people investing on their own, from large institutions, and corporations. However, in contrast to more established commodities exchanges, the cryptocurrency market is more volatile. It is unstable and gains are uncertain, and unexpected because of the various technological, emotional, and legal aspects that might affect it. Therefore, the quantities in the columns have shown to increase in the long term, but have shown ups and downs at short intervals of time, often unrelated, due to external market circumstances and/or market sentiments over the years. The dataset is prepared from the data provided by Tronscan [46] website which manages the entire data of Tron cryptocurrency and ecosystem. We use the daily price which has high, low, and, day prices and the data containing the volumes of trades. We scale all our data to make the data of the same order (say between 1 to 10) using the standard scaler standardization technique. We use python to merge the uniform data for the same days for transaction success prediction. We chose the data carefully to obtain a better correlation and use relevant features that are useful in a market-based analysis. Features relevant to regression are concatenated, and a new dataset is created, merging the price dataset and the related dataset.

Figure 1 shows the heatmap showing the feature correlation. Some of the features show significant growth of the cryptocurrency and directly relate to strengthening the feature mentioned above and the scope of TRON. The daily price provides insights to the TRX coin market trends in short-term and prevailing market sentiments. The features such as ‘totalTrxTransfer’ and ‘totalTransactions’ refer to the transactions within the TRON environment, similar to the other features in Figure 1. In regression analysis, these features are useful in establishing correlation between different variables.

![Figure 1. The plot for transaction volume correlation of Tron price and transactions.](image-url)
The relation between the values are plotted for the different features relating to the amount of Tron transferred from mid-2018 to mid-2022 and it is represented in Figures 2 and 3. In Figure 2 the $x$-axis represents number of days and $y$-axis represents the numerical values of these features. This shows that market cap and volume have really high values compared to the rest of the features. Though there is a wealth of literature on cryptocurrency price prediction, only some methods are practical for use in real-time. Independent validation is therefore seen at a time of unexpected upheaval, but little difference in prices occurs during a period of bearish markets; this allows us to evaluate the accuracy of the predictions regardless of whether or not the market trend reverses between the two periods.

**Figure 2.** The plot for the volume and market features correlation.

**Figure 3.** The pair-plot for market functions feature correlation for TRX Token.
The use of cryptocurrencies as a novel medium for executing secure monetary transactions and storing value has grown in prominence in recent years. Due to the transparency of cryptocurrency transactions, quantitative analyses of many characteristics of virtual currencies are feasible. A candlestick chart condenses data from many time frames into a uniform price bar. They are, therefore, more beneficial than conventional open, high, low, and close (OHLC) bars or straight lines that connect closing price dots. The red candles represent fall in price and green candles represent the rise in prices. The candlestick chart for TRON is shown in Figure 4.

Figure 4. The fluctuations of the price of TRX coin for a specific period of time.

Figure 5 shows a 30-day moving average, which represents the closing price and the 30-day average price for four years. It is very clear from the plot that cryptocurrency, and in our case TRON has shown significant volatility over the 30 day average in the past, therefore we wish to help users make informed decisions for the long term but analysis would be beneficial to make these decisions (Figures 6 and 7).

Figure 5. The plot for 30 day moving average of Tron price and transactions along with the closing price for a certain period.
Figure 6. The candle plot for moving average chart for the closing price for four years.

Figure 7. The detailed flow of proposed methodology.
We propose an effective framework for potentially predicting future prices given the wide scope of TRON cryptocurrency and discuss aspects that have been neglected before. The aim of studies such as this is to adapt to changing circumstances of the volatile nature of cryptocurrencies and the financial risks involved. The whale-optimized algorithm (WOA) is introduced along with ensemble classifiers, which can be called to train the models and more often results in improved accuracy and performance. The WOA algorithm is applied to obtain the optimized value and make a prediction on the transaction values. The Whale optimization algorithm is used for selecting the most optimal features for both of the predictions along with principal component analysis to potentially reduce the dimensionality of the data if applicable for the given scenario. The Returns column indicates the percentage increase in the invested capital.

\[\text{Returns}_i = \frac{\text{Close}_i - \text{Open}_i}{\text{Open}_i}\]  

A crucial parameter is the dayChange, which represents the fluctuations in the market on a day-to-day basis. Essentially, it is the difference between the closing prices of two consecutive days. Under certain circumstances, it may be given approximately by the modulus of difference of closing and opening prices. This is relevant for the closing price of the next day, calculating which is essential for price prediction.

\[\text{dayChange}_i = \text{Close}_i - \text{Close}_{i-1}\]  

OR \[
\text{dayChange}_i = \text{Close}_i - \text{Open}_i
\]

Changes in the daily prices can be expressed as shown in Equation (6). A binary variable CallOrder represents if trading on a specific day would result in a gain or loss, which can be derived by applying the unit step function to the DayChange.

\[\text{callOrder}_i = \text{Heaviside}(\text{dayChange}_i)\]  

Likewise, the Equation (8) gives closing price of a cryptocurrency for a complex OHLC dataset.

\[\text{Close}_i = f_{\text{Simple}}(\text{Close}_{i-1})\]  

Now coming to the WOA optimization, we assume the actual best solution to be near the target fish (the solution) in an ocean and the other objects update their positions across the most optimal agent,

\[\vec{D} = |\vec{C} \cdot \vec{P}_{\text{best}}(t) - \vec{P}(t)|\]  

In this context, “t” represents the current cycle, and the position vector of the humpback whale during its hunting strategy is denoted by \(\vec{P}\). The best solution is represented by \(X^*_s\), while coefficient vectors are denoted by \(\vec{A}\) and \(\vec{C}\). Essentially, \(\vec{P}\) refers to the location of the humpback whale. This phase is often called encircling the fish. \(\vec{a}\) represents the coefficient vector, which is used to control the exploration and exploitation rates of the algorithm, and \(\vec{A}\) is the updated coefficient vector.

\[\vec{P}(t+1) = \vec{P}_{\text{best}}(t) - \vec{A} \cdot \vec{D}\]  

\[\vec{A} = 2\vec{a} \ast \vec{K}_1 - \vec{a}\]  

Regarding the bubble-net attack stage, which is sometimes referred to as the exploitation stage, we utilize the process of shrinking to demonstrate how humpback whales surround their prey. As the iterations progress, the value of a certain variable, denoted as \(\vec{C}\), is gradually reduced from 2 down to 0.
\[ \vec{C} = 2k_2 \]  

Here, \( \vec{F}_1 \) and \( k\vec{2}_2 \) are random vectors in the range of \([0,1]\). Variable \( m \) is used in the cosine function. Its value affects the direction and magnitude of the steps taken towards the best solution. For the spiral updating position, the following functions are incorporated,

\[ \vec{D}' = |\vec{P}_{\text{best}}(t) - \vec{P}(t)| \]  

\[ \vec{P}(t + 1) = \vec{D}' \cdot \cos(2\pi m) + \vec{P}_{\text{best}}(t) \]  

In order for the humpbacks to update their position, the whales randomly search with respect to the position of their counterparts.

\[ \vec{D} = |\vec{C} \cdot \vec{P}_{\text{rand}}(t) - \vec{P}(t)| \]  

\[ \vec{P}(t + 1) = \vec{P}_{\text{rand}}(t) - \vec{A} \cdot \vec{D} \]  

Equation (17) calculates the coefficient \( a \) as iteration \( t \) progresses from 1 to \( T \).

\[ a = 2 - t \times \frac{2}{T} \]  

Equation (18) computes the center of the search space \( C \) by averaging positions of agents \( X_i \).

\[ C = \frac{1}{N} \sum_{i=1}^{N} X_i \]  

Equation (19) calculates the distance \( D_i \) between the current position \( X_i \) of agent \( i \) and the best position \( A \) found so far, scaled by the coefficient \( a \).

\[ D_i = |C - a \times |A - X_i|| \]  

Equation (20) computes the parameter \( b \) used for updating the position of agents based on the random values \( r_1 \) and \( r_2 \).

\[ b = \frac{-1}{\log(r_2 \times \frac{A - X_i}{D_i})} \]  

Equation (21) updates the position of agent \( i \) by moving it towards the best position \( A \) using a spiral-shaped movement when \( r_1 < 0.5 \).

\[ X_i \leftarrow A - D_i \times e^{-b \times r_2} \times \cos(2 \times \pi \times r_2) \]  

Equation (22) updates the position of agent \( i \) by moving it away from the best position \( A \) using a spiral-shaped movement when \( r_1 \geq 0.5 \).

\[ X_i \leftarrow A + D_i \times e^{-b \times r_2} \times \cos(2 \times \pi \times r_2) \]  

Equation (23) calculates the R-squared score \( r_{\text{best}} \) as the final performance metric for the optimized regression model. \( D_{\text{test}} \) refers to the test data.

\[ r_{\text{best}} \leftarrow r^2_{\text{score}}(D_{\text{test}}, \hat{y}) \]  

Equation (24) calculates the distance between each agent and the best solution found so far.

\[ D \leftarrow |C \times A - X \times A| \]
Equation (25) updates the position of each agent according to the Whale Optimization Algorithm.

\[ X' \leftarrow A - p \times D \quad (25) \]

Equation (26) predicts the class labels \( y_{\text{pred}} \) for the test data \( x_{\text{test}} \) using the optimized classifier \( clf \).

\[ y_{\text{pred}} = clf.predict(x_{\text{test}}) \quad (26) \]

The Equation (27) calculates the accuracy of the classifier by dividing the number of correct predictions by the total number of predictions made.

\[ \text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \quad (27) \]

Equation (28) calculates the precision of the classifier, which is the proportion of true positive predictions out of all positive predictions made.

\[ \text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (28) \]

Equation (29) calculates the recall of the classifier, which is the proportion of true positive predictions out of all actual positive instances in the dataset.

\[ \text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (29) \]

Equation (30) computes the F1 score, which is the harmonic mean of precision and recall. The F1 score is a popular metric used to evaluate classifiers when dealing with imbalanced datasets.

\[ \text{F1\_Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (30) \]

In Equation (31) \( C \) is the confusion matrix, \( y_{\text{test}} \) are the true class labels, and \( y_{\text{pred}} \) are the predicted class labels. The confusion matrix \( C \) is a square matrix of size \( k \times k \), where \( k \) is the number of classes. Each element \( C_{ij} \) represents the number of instances of class \( i \) that were classified as class \( j \).

\[ C = \text{confusion\_matrix}(y_{\text{test}}, y_{\text{pred}}) \quad (31) \]

In Algorithm 1, first, the population of profit classifications, denoted as \( P_1 \), is initialized. The training sets are split into \( n \) folds using Repeated Stratified K-fold cross-validation. The first \( n-1 \) folds are used as the base model, and the predictions are made for the \( n \)-th fold. The predictions from steps 1, 2, and 3 are concatenated into the \( x_{1\_train} \) list. The list is converted to its binary representation and the fitness of each solution in the population is computed followed by initialisation of best solution as \( P^* \). After processing each solution, the algorithm checks if any search agent has moved beyond the search space and corrects the position if necessary. The list is again converted to its binary representation and the fitness of each search agent is computed. The iteration counter \( t \) is incremented. Finally, the algorithm returns the best solution \( P^* \). All the relevant equations are explained in Equations (9)–(16) related to this algorithm. This is the general algorithm for implementation of Whale optimization algorithm and is applicable for similar cryptocurrency datasets (or a retrieved OHLC data). It is worth noting that using \( A = 1 \) can lead to slower convergence to the global optimum, as it does not allow for larger jumps in the search space. It may be beneficial to use larger values of \( A \) to allow for more exploration of the search space. At each iteration of the algorithm, a new set of candidate parameter values is generated using a combination of random mutation and cross-over (i.e., taking the best parts of two or more candidate solutions and combining them). These new solutions are then evaluated using the objective function, and the best ones are kept as the basis for the
next iteration. This gives us the best value of the hyper-parameters to then train our model. The algorithm converges on a set of parameter values that produce a high-performing classification model, as measured by the objective function. The algorithm may need to be run several times with different starting parameters in order to find the best possible solution. A good number of iterations is 100 in general, though the algorithm in itself is very complex, and more almost always lead to having a higher probability of finding a better solution.

Algorithm 1 Optimal Global Position Vector

1. procedure OPTIMALGLOBALPOSITION(I, P)
2. Initialize the population $P_i (i = 1, 2, 3, \ldots, N)$ of profit classification.
3. Split the training sets into $n$ folds using Repeated Stratified K-fold.
4. Take the first fold of the base model as $n - 1$, $n$-th folds are to be predicted.
5. The predictions using steps 1, 2, and 3 are concatenated to the $x1_{train}$ list.
6. Convert the list to binary representation.
7. Compute the fitness of each solution.
8. Set the best solution as $P^*$. 
9. while $(t < I)$ do
   10. for each solution do
      11. if $(p < \frac{1}{2})$ then
      12. The particle position is updated by $\vec{D} = |\vec{C} \odot \vec{P}_{rand}(t) - \vec{P}(t)|$
      13. else
      14. if $(|A| < 1)$ then
      15. Select a random particle $P_{rand}$.
      16. Update the particle position by $\vec{P}(t + 1) = \vec{P}_{rand}(t) - \vec{A} \odot \vec{D}$.
      17. end if
      18. end if
   19. end if
20. end for
21. Investigate whether any search agent moves beyond space and correct the sample.
22. Convert the list to binary representation.
23. Compute the fitness of every search agent.
24. Update $P^*$ if there is a better solution.
25. $t = t + 1$.
26. end while
27. return $P^*$
28. end procedure

Algorithm 2, titled “Whale Optimization Algorithm for Regression”, aims to optimize the regression models (e.g., LightGBM, XGBoost, CatBoost) for achieving the best R-squared score using the Whale Optimization Algorithm. The algorithm takes as input the training data $D_{train}$, testing data $D_{test}$, a set of regression models $M$, a search space $S$, the number of agents $N$, and the number of iterations $T$. The main objective of the algorithm is to find the best combination of hyperparameters for the chosen regression model.

The FitnessFunction is defined to measure the performance of a solution by fitting the model to the training data and predicting the target variable on the test data. The fitness function returns the negative R-squared score to guide the optimization process. All the equations related to Algorithm 2 are explained in Equations (17)–(23) and (32)–(34).
Algorithm 2 Whale Optimization Algorithm for Regression

**Require:** Training data $D_{\text{train}}$, testing data $D_{\text{test}}$, regression models $M$ (e.g., LightGBM, XGBoost, CatBoost), search space $S$, number of agents $N$, number of iterations $T$

**Ensure:** Best R-squared score $r_{\text{best}}$ found

1: function FITNESSFUNCTION($X$, $M$, $D_{\text{train}}$, $D_{\text{test}}$)
2:     $x_{\text{train}}$ ← DataFrame($X$, columns=$D_{\text{train}}$.columns)
3:     $M$.fit($x_{\text{train}}$, $D_{\text{train}}$.target)
4:     $\hat{y}$ ← $M$.predict($D_{\text{test}}$)
5:     return $-r^2_{\text{score}}(D_{\text{test}}\cdot\text{target}, \hat{y})$
6: end function

7: $X$ ← RandomUniform($\text{low} = S_{\text{low}}, \text{high} = S_{\text{high}}, \text{size} = (N, S_{\text{dim}})$)
8: $A$ ← None
9: $r_{\text{best}}$ ← $-\infty$
10: for $t = 1$ to $T$ do
11:     $a$ ← $2 - t \cdot \frac{2}{T}$
12:     $C$ ← $\frac{1}{N} \sum_{i=1}^{N} X_i$
13:     for $i = 1$ to $N$ do
14:         Clip $X_i$ to $S$
15:         $f_i$ ← FITNESSFUNCTION($X_i$, $M$, $D_{\text{train}}$, $D_{\text{test}}$)
16:         if $f_i > r_{\text{best}}$ then
17:             $r_{\text{best}}$ ← $f_i$
18:             $A$ ← $X_i$
19:         end if
20:     $D_i$ ← $||C - a \times |A - X_i||$
21:     $r_1, r_2$ ← RandomUniform(0, 1)
22:     $b$ ← $\frac{-1}{\log(r_2 \cdot \frac{r_1}{r_2})}$
23:     if $r_1 < 0.5$ then
24:         $X_i$ ← $A - D_i \cdot e^{-b \cdot r_2} \times \cos(2 \times \pi \times r_2)$
25:     else
26:         $X_i$ ← $A + D_i \cdot e^{-b \cdot r_2} \times \cos(2 \times \pi \times r_2)$
27:     end if
28: end for
29: end for
30: $x_{\text{train\_best}}$ ← DataFrame($A$, columns=$D_{\text{train}}$.columns)
31: $M$.fit($x_{\text{train\_best}}$, $D_{\text{train}}$.target)
32: $\hat{y}$ ← $M$.predict($D_{\text{test}}$)
33: $r_{\text{best}}$ ← $r^2_{\text{score}}(D_{\text{test}}\cdot\text{target}, \hat{y})$
34: return $r_{\text{best}}$

The algorithm initializes the positions of $N$ agents in the search space and iterates for a given number of iterations $T$. In each iteration, the algorithm updates the agents’ positions according to the Whale Optimization Algorithm equations, which involve calculating the distance between each agent and the best solution found so far. The agents’ positions are updated based on random variables and the distances calculated. The agents’ positions are clipped to the search space to ensure they remain within the search space boundaries. The fitness of each agent is evaluated, and the best solution is updated if a better one is found. The R-squared score is calculated for the best solution at the end of the algorithm, and the best R-squared score is returned.

The Algorithm 3 uses various variables, including $D_{\text{train}}$ and $D_{\text{test}}$, which represent the training and testing data, respectively. The variable $M$ represents a set of classification models, such as Logistic Regression and Random Forest. The search space for the optimization process is denoted by $S$, while the number of agents in the search space is represented by $N$. The algorithm runs for a specified number of iterations, indicated by $T$. The positions of agents within the search space are stored in $X$. The position of the best solution found so far is represented by $A$, and the best accuracy score achieved is denoted by $a_{\text{best}}$. The
The equations used in Algorithm 3 are explained in Equations (17), (18), (24), (25) and (32)–(34). \( S_{\text{dim}} \) refers to the dimensions of the desired output shape for the array generated by the ‘RandomUniform’ function, containing uniformly distributed numbers between 0 and 1. The notation “:-1” is a slicing notation which means “all elements from the start to one before the end”.

**Algorithm 3 Whale Optimization Algorithm for Classification using Stacking and Voting Classifiers**

**Require:** Training data \( D_{\text{train}} \), testing data \( D_{\text{test}} \), classification models \( M \) (e.g., Logistic Regression, Random Forest, K-Nearest Neighbors, Gaussian Naive Bayes, Stochastic Gradient Descent, Decision Tree, Extra Trees), search space \( S \), number of agents \( N \), number of iterations \( T \)

**Ensure:** Best accuracy score \( a_{\text{best}} \) found for the classifiers

1. **function** \( \text{FITNESSFUNCTION}(X, M, D_{\text{train}}, D_{\text{test}}) \)
2. \( x_{\text{train}} \leftarrow \text{DataFrame}(D_{\text{train}}.iloc[:, :-1], \text{columns}=D_{\text{train}}.columns[:-1]) \)
3. \( x_{\text{test}} \leftarrow \text{DataFrame}(D_{\text{test}}.iloc[:, :-1], \text{columns}=D_{\text{test}}.columns[:-1]) \)
4. \( y_{\text{train}} \leftarrow D_{\text{train}}.\text{target} \)
5. \( \text{estimators} \leftarrow [] \)
6. for \( m \in M \) do
7. \( \text{pipe} \leftarrow \text{Pipeline}([(m, m())]) \)
8. \( \text{estimators}.\text{append}((m\ .\text{class} .\text{name}, \text{pipe})) \)
9. end for
10. \( \text{voting} \leftarrow \text{VotingClassifier(}	ext{estimators=estimators, voting='soft'}) \)
11. \( \text{stacking} \leftarrow \text{StackingClassifier(}	ext{estimators=estimators, final_estimator=voting}) \)
12. \( \text{stacking} .\text{fit}(x_{\text{train}}, y_{\text{train}}) \)
13. \( \text{score} \leftarrow \text{stacking.score}(x_{\text{test}}, D_{\text{test}}.\text{target}) \)
14. return \( \text{score} \)
15. end function
16. \( X \leftarrow \text{RandomUniform}(\text{low}=\text{Slow}, \text{high}=S_{\text{high}}, \text{size}=(N, S_{\text{dim}})) \)
17. \( A \leftarrow \text{None} \)
18. \( a_{\text{best}} \leftarrow 0 \)
19. for \( t = 1 \) to \( T \) do
20. \( a \leftarrow 2 - t \times \frac{2}{T} \)
21. \( C \leftarrow \frac{1}{N} \sum_{i=1}^{N} X_{i} \)
22. for \( i = 1 \) to \( N \) do
23. Clip \( X_{i} \) to \( S \)
24. \( f_{i} \leftarrow \text{FITNESSFUNCTION}(X_{i}, M, D_{\text{train}}, D_{\text{test}}) \)
25. if \( f_{i} > a_{\text{best}} \) then
26. \( A \leftarrow X_{i} \)
27. \( a_{\text{best}} \leftarrow f_{i} \)
28. end if
29. end for
30. \( p \leftarrow \text{RandomUniform(size}=S_{\text{dim}}) \)
31. \( D \leftarrow | C \times A - X \times a | \)
32. \( X' \leftarrow A - p \times D \)
33. end for
34. \( x_{\text{train}_{\text{est}}} \leftarrow \text{DataFrame}(A, \text{columns}=D_{\text{train}}.\text{columns}[:-1]) \)
35. \( \text{stacking.fit}(x_{\text{train}_{\text{est}}}, D_{\text{train}}.\text{target}) \)
36. \( a_{\text{best}} \leftarrow \text{stacking.score}(x_{\text{test}}, D_{\text{test}}.\text{target}) \)
37. return \( a_{\text{best}} \)

The methodology of applying the algorithm consists of three steps:

1. Applying the Regression Techniques: Gradient boosting regressors have been very successful in price prediction-based regression. XGBoost, CATBoost, and LGBM regressors are used for dynamic pricing estimation and we compare the efficiency of these regressors for our prepared TRX dataset. The tree-based regressors such as
XGBoost, CatBoost, and LightGBM are useful in regressors have feature selection capabilities built-in that help identify the most important features for the regression task. The libraries are useful regularization techniques to prevent overfitting. XGBoost, CatBoost, and LightGBM can still be significantly faster than other algorithms when used along with a meta-heuristic.

2. Classification using standard classifiers; After the regression, we move on to the classification of profit and loss using different classifiers. We divide the data into training and testing sets and the training data are used to train each model simultaneously.

3. Classification using an ensemble of classifiers; The next stage is to analyze the forecasts of the models. Accuracy may be improved by ensemble techniques with XGBoost when dealing with asymmetric data. We first focus on classifying profits and losses to check if there is a profit, followed by an inspection of the accuracy of transaction success & failure considering TRX transactions over Blockchain. Equations (27)–(31) give more explanation about the evaluation metrics used.

XGBoost, CATBoost, and LGBM regressors are used for the regression. Table 1 shows the different metrics for regressions performed by the models. Figure 8 shows the scatter plots for corresponding (a). XGBoost regressor, (b). LGBM regressor and (c). CATBoost Regressor. A higher R² score generally indicates that the model is a better fit to the data. It is given by

\[ R^2 = 1 - \frac{SSres}{SStot} \]

where SSres is the squared residuals (the difference between the predicted and actual values) and SStot is the total sum of squares (i.e., the differences between the actual values and the mean of the target variable).

<table>
<thead>
<tr>
<th>Regressor Model</th>
<th>R Squared</th>
<th>RMSE</th>
<th>MAE</th>
<th>Explained Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>XGBoost Regressor</td>
<td>0.982032</td>
<td>0.004142</td>
<td>0.002027</td>
<td>0.982040</td>
</tr>
<tr>
<td>CATBoost Regressor</td>
<td>0.985809</td>
<td>0.003775</td>
<td>0.002028</td>
<td>0.984824</td>
</tr>
<tr>
<td>LGBM Regressor</td>
<td>0.982514</td>
<td>0.004003</td>
<td>0.001916</td>
<td>0.982536</td>
</tr>
</tbody>
</table>

Figure 8. The plot displays scatterplots for XGboost, LGBM Regressor and CATBoost, respectively, for the TRON dataset.

4. Results and Discussion

The x-axis of each scatterplot shows the true values of the next day’s closing price, while the y-axis shows the predicted values of the next day’s closing price. The scatterplots are useful for visualizing how well the models perform in predicting the next day’s closing price of Tron. The closer the scatterplot points are to the line of perfect predictions (y = x), the better the model is at predicting the next day’s closing price of Tron.

We then move to the classification problem, which helps with determining whether the cryptocurrency is going to make a profit or loss the next day based on the historical performance of the TRX coin. It is essential to consider that the price of any cryptocurrency, as noted in the literature review, can fluctuate significantly depending on factors that impact the market and factors inherent to the nature of the blockchain technology. For our classification problems, we analysed the performance of several models and observed that tree based models perform the best when it comes to smaller datasets. It is reflected in our analytics by the XGBoost classifier, which uses a tree-based gradient boosting algorithm.
It is important to reduce its complexity to minimize any overfitting, which is effectively performed using L2 regularization and manual tuning while not including these hyperparameters when defining the search space. The voting classifier takes several tree-based classifiers as inputs along with the other models, similar to stacking the classifier but in a more effective way.

We found that the accuracy of the models such as Random Forest, Decision Tree, Gaussian Naive Baise, Logistic Regression, K-Nearest Neighbors, Scholastic Gradient Descent is on the lower end when compared to these ensemble techniques. The stacking classifier uses all these models plus the gradient boosters as pipelines with the final estimator being the voting classifier. Hyperparameter optimization then sets the ideal settings for learning rate, booster type, maximum depth, number of estimators, and other factors classifiers in our tabular dataset. From the standard techniques, XGBoost provides the best accuracy because of its superior performance in tabular datasets and tree-based structures. The corresponding confusion matrix before applying WOA is shown in Figure 9. The use of a meta-heuristic whale optimization algorithm with staking classifier tends to bypass local optima.

![Confusion matrices of first 25 epochs of profit prediction classification](image)

*Figure 9. Confusion matrices of first 25 epochs of profit prediction classification (a) SGD (b) LR (c) LR (d) Random Forest (e) KNN and (f) DT classifier.*

In Algorithm 1, Equation (32) initializes the positions of the agents in the search space $X$ with random values uniformly distributed between the lower and upper limits of the search space ($S_{\text{low}}$ and $S_{\text{high}}$), and with a size of $(N, S_{\text{dim}})$.

$$X \leftarrow \text{RandomUniform}(\text{low} = S_{\text{low}}, \text{high} = S_{\text{high}}, \text{size} = (N, S_{\text{dim}})) \quad (32)$$

Similarly, Equation (33) gives random numbers in the range $[0, 1]$ used for updating the positions of agents.

$$r_1, r_2 \leftarrow \text{RandomUniform}(0, 1) \quad (33)$$

Equation (34) creates a new DataFrame named $x_{\text{train\_best}}$ using the best agent’s position $A$. The DataFrame is created with the same column names as the training data $D_{\text{train}}$. This is
performed to ensure that the data structure is consistent with the input format required by
the regression models.

\[ x_{\text{train\_best}} \leftarrow \text{pd.DataFrame}(A, \text{columns} = D_{\text{train\_columns}}) \]  (34)

Comparing the models, CatBoost seems to perform slightly better than the other two mod-
els, with the highest R-squared value, lowest MSE and RMSE, and a slightly higher median
absolute error (MedAE) than LGBM. However, the differences in performance are relatively
small, and all of the models perform quite well. Techniques such as regularization or early
stopping are useful and help prevent overfitting to a certain extent. The optimization
algorithm does a good job in training the model in a more efficient and effective manner,
which could indirectly help in reducing overfitting. However, reducing the complexity of
the model by not considering the parameters which lead to a more complex model is a good
approach to tackle the problem of overfitting along with regularization and cross-validation.
Although the SGD classifier gets no true positive or false negative. This situation often
arises when the model predicts only one class for all instances. The WOA addressed this by
correct hyperparameters, with the results as shown in graphs and tables. Now, the prophet
forecaster shows the prediction for future dates based on the current dates in the dataset.
It is depicted in Figure 10. Table 2 shows some of the hyper-parameters that are included
in the search to be tuned during the training of these models. It is important to note that
n_estimators, reg_alpha or reg_lambda, min_data_in_leaf, min_gain_to_split should be
preferred to be manually tuned to avoid overfitting, and any other hyperparameter should
be accessed for overfitting on a case by case basis. Cross-validation (5, 8 or 10 folds) must
be performed for accessing if there is overfitting and lower performance on testing data
with respect to the corresponding training data.

![Figure 10](image)

**Figure 10.** Prediction of the future price of TRON using Prophet Forecaster. Black dots represent
historical TRON prices, blue line represents forecast and shaded area represent forecast uncertainty.

We then move to the classification problem, which helps with determining whether
the cryptocurrency is going to make a profit or loss the next day based on the historical
performance of the TRX coin. It is essential to consider that the price of any cryptocurrency,
as noted in the literature review, can fluctuate significantly depending on factors that
impact the market and factors inherent to the nature of the blockchain technology.
Table 2. Most important Hyper parameters considered for the training purpose, only relevant hyper parameters are searched using WOA.

<table>
<thead>
<tr>
<th>Regressor Model</th>
<th>Hyper Parameters Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>XGBoost Regressor</td>
<td>subsample, eta, booster, min_child_weight, learning_rate colsample_bytree, validate_parameters, and nthread</td>
</tr>
<tr>
<td>CATBoost Regressor</td>
<td>subsample, random_strength, learning_rate, l2_leaf_reg, colsample_bylevel, bagging_temperature, and boosting_type</td>
</tr>
<tr>
<td>LGBM Regressor</td>
<td>subsample, learning_rate, colsample_bytree, max_bin, objective, min_data_in_leaf, and feature_fraction</td>
</tr>
</tbody>
</table>

For our classification problems, we analysed the performance of several models and observed that tree based models perform the best when it comes to smaller datasets. It is reflected in our analytics by the XGBoost classifier, which uses a tree-based gradient boosting algorithm. It is important to reduce its complexity to minimize any overfitting, which is effectively performed using L2 regularization and manual tuning while not including these hyper-parameters when defining the search space. The voting classifier takes several tree-based classifiers as inputs along with the other models, similar to stacking the classifier but in a more effective way. We found that the accuracy of the models such as Random Forest, Decision Tree, Gaussian Naive Baise, Logistic Regression, K-Nearest Neighbors, Scholastic Gradient Descent is on the lower end when compared to these ensemble techniques. The stacking classifier uses all these models plus the gradient boosters as pipelines with the final estimator being the voting classifier. Hyperparameter optimization then sets the ideal settings for learning rate, booster type, maximum depth, number of estimators, and other factors classifiers in our tabular dataset. From the standard techniques, XGBoost provides the best accuracy because of its superior performance in tabular datasets and tree-based structures. The corresponding confusion matrix of the model is shown in Figure 11. The use of a meta-heuristic whale optimization algorithm with staking classifier tends to bypass local optima.

![Figure 11](image-url)
This is evident from the results that the base model has distinct performance over the dataset for the next day prediction. The dataset is relatively small, so the accuracy of these base models is found to be quite close to each other. The dataset for price prediction is quite balanced; however, the test set does not have many values to make accurate predictions. This is not much of a problem since tree-based models perform well even on this small dataset and perform exceedingly well when used in stacking and voting modules. The training and testing are first carried out using different machine learning classifiers with a test-train split of 30–70; next, we use it with the proposed stacking and the voting-based ensemble models. Random Forest, Support Vector Classifier, Decision Tree Classifier, ExtraTrees Classifier, Light Gradient Boosting Classifier, XGBoost, and Logistic Regression Classifier are some of the classifiers used by the stacking model. Each of these models provided appreciable accuracy when used alone, but the number of false negatives and false positives varied greatly. This is subsequently reflected in the recall graph.

Each of these individual classifiers may have a unique approach or strategy for analyzing the dataset, which can be seen from the results. However, simply taking the majority decision may not always be the best approach, because some classifiers may give more false-positives or true-negatives in classification others. In such cases, it might be better to consider all the factors related to a classifier and the algorithm behind it before using it for predictions. In a general sense, the WOA algorithm found a set of hyperparameters that increase the number of trees in the ensemble model, (n_estimators) and limit the depth of each tree (max_depth) to avoid overfitting. It also adjusts the learning rate (learning_rate) to optimize the trade-off between the speed of convergence and the accuracy of the model. Figure 12 shows the accuracies obtained by the models for both of our classification problems where we use whale optimization algorithms. The confusion matrices for the stacking classifier are shown in Figure 11 for both the tasks i.e., profit and success rate prediction.

![Accuracy analysis of the models for profit classification and transaction success rate.](image)

The other classification system is used to determine if machine learning can predict success and failure rates along with the profits to cover another aspect of the real-world scenario for the cryptocurrency with good accuracy. In this case, we consider data such as sender, receiver, transaction type, status, amount and token type. The Status could be confirmed and unconfirmed in this case to address records as SUCCESS or FAILED (Table 3).
Table 3. The average accuracy of the models over several runs for profit classification and transaction success rates.

<table>
<thead>
<tr>
<th>Machine Learning Model</th>
<th>Profit Classification Accuracy (%)</th>
<th>Transaction Success Rate Classification Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic Regression (LR)</td>
<td>52.278</td>
<td>86.94</td>
</tr>
<tr>
<td>Random Forest (RF)</td>
<td>63.3</td>
<td>87</td>
</tr>
<tr>
<td>K-Nearest Neighbour (KNN)</td>
<td>51.502</td>
<td>87</td>
</tr>
<tr>
<td>Gaussian Naive Bayes (GNB)</td>
<td>53.433</td>
<td>86.5</td>
</tr>
<tr>
<td>Stochastic Gradient Descent</td>
<td>50.54</td>
<td>78.04</td>
</tr>
<tr>
<td>Decision Tree (DT)</td>
<td>71.89</td>
<td>86.93</td>
</tr>
<tr>
<td>Extra Tree (ET)</td>
<td>55.79</td>
<td>86.97</td>
</tr>
<tr>
<td>XGBoost (XGB)</td>
<td>80.042</td>
<td>90.54</td>
</tr>
<tr>
<td>Voting Classifier</td>
<td>84.55</td>
<td>97.21</td>
</tr>
<tr>
<td>Stacking Classifier</td>
<td>89.05</td>
<td>98.88</td>
</tr>
</tbody>
</table>

The proposed work focuses on enhancement in deployment mechanisms and superior optimization techniques to predict the success and failure of transactions in a TRON environment. All the reported values are mean values for 5 runs, and this is performed to ensure reduced bias of the models towards optimized solutions. One thing to note, however, is that it’s best to reset the environment after each run in order to reduce re-fitting towards optimal values in a particular run.

Figures 13–15 show the F1, scores (the harmonic mean of precision and recall), precision and recall values, respectively. These scores show that some models consistently better than the others and have higher scores than the others for all the metrics. Stacking classifier has the best overall performance across all the metrics. Ensemble models should always be assessed for any overfitting at all times when used for unbalanced and smaller dataset and regularisation techniques that we applied are useful of avoiding overfitting to a certain extent.

![Figure 13. F1 values for profit or loss classification and transaction success rates.](image)
There is very limited research on TRON to compare our results and specifically on the dataset that we use. It is concluded in the research work by Yadav et al. [47] in 2021 that TRON performs the best out of five cryptocurrencies in their analysis. Similarly, research by Malsa et al. [48] shows that the TRX coin has a promising future based on their qualitative and quantitative analysis of the parameters of the coin. From recent research performed for 10 different cryptocurrencies on a time-series dataset [49], TRON shows promising results compared to similar tokens, obtaining an MSE value ranging from 0.0007 to 0.0009, RMSE ranging from 0.0081 to 0.0094 and MAE value ranging from 0.0052 to 0.0061.
Our models perform better for most of the values, potentially due to a robust hyperparameter tuning methodology combined with using tree-based regressors which perform significantly better on smaller datasets [50]. It is important to note that even time series data may contain similar OHLC data that have significant overlap with our dataset. One important analysis from the literature on crypto price prediction is that the newer cryptocurrencies tend to show better performance when compared to their established counterparts in terms of MAE, R squared, and RSME values. Our findings show similar results and they reflect in other research works as well.

The models with the highest precision values for this binary classification problem are Stacking and Voting, with precision values ranging from 85.19% to 99.84%. Logistic Regression (LR) and XGBoost (XGB) also performed consistently well, achieving high precision values in addition to high overall accuracy.

Interestingly, LR and KNN show abnormally high values of recall for transaction success rate classification. Another possibility is that it is due to the imbalanced dataset itself, meaning that there are more negative cases than positive cases. One possibility is that the model is able to learn relevant patterns in the data that distinguish positive cases from negative cases. For example, in a binary classification problem, the model may learn to recognize specific features or combinations of features that are more likely to be present in positive cases. If these patterns are strong and consistent across the dataset, the model may be able to correctly identify a high proportion of the actual positive cases, leading to a high recall value.

For all the algorithms executed, a pipeline is created using StandardScaler(), PCA(), and the corresponding classifier. The StandardScaler is used to normalize the input data, and PCA is used for dimensionality reduction. The Stacking classifier takes the pipelines as input, and the final estimator is a Voting Classifier that takes the classifiers as input. In this case, the regression models are trying to predict the next closing price of TRON based on historical data. The input data includes various features (such as the opening price, high price, low price, and volume) from past trading periods, and the output is the next closing price of TRON.

Since we want to optimize the hyperparameters of the voting and stacking classifiers, we treat this as a continuous optimization problem. In the algorithm for optimal global position vector, the position of each agent represents a solution in the search space. The algorithm starts by initializing the population of agents randomly in the search space. Then, for each agent in the population, the position is updated based on the agent’s current position and the position of the best agent found so far. The updated equations for the position depends on the current iteration, which determines the type of search being performed (exploration or exploitation). The fitness of each agent is evaluated using the fitness function. If the fitness of an agent is better than the fitness of the best agent found so far, the best agent is updated. The step size is also updated in each iteration. The algorithm stops after a maximum number of iterations is reached, and the position of the best agent found is returned as the optimal solution.

For the regression, The XGBRegressor and LightGBM models use similar boosting algorithms but differ in their implementation. The XGBRegressor uses a pre-sorted algorithm to build trees, while the LightGBM uses a histogram-based algorithm. The CatBoostRegressor is another gradient-boosting model that uses ordered boosting and a random permutation to train the decision trees. Once trained, the models can be used to make predictions on new data (in this case, the test data) by taking in the input features and outputting a predicted closing price. The accuracy of the model is typically measured using a performance metric such as mean squared error or R-squared, which is best to use in our case. The hyperparameters that tuned for the XGBoost, LightGBM, and CatBoost regressors include but are not limited to the learning rate, number of estimators, maximum depth, gamma, subsample ratio, and regularization parameters. The search space for the WOA can be defined by setting the bounds for each hyperparameter to be tuned. The fitness function for the WOA is set to evaluate the performance of the model with the given hyperparameters.
This is performed by training the model with the given hyperparameters and evaluating its performance on a validation set. The validation set can be used to prevent overfitting during hyperparameter tuning.

The voting classifier uses these XGB, LGBM, and CatBoost models for voting, with 6 other models as estimators. These models are: XGBClassifier with n_estimators = X[0] and max_depth = int(X[1]), CatBoostClassifier with learning_rate = X[2], and LGBMClassifier with learning_rate = X[3]. The three models are combined in an ensemble model using a VotingClassifier with a ‘soft’ voting strategy. The accuracy of the model is computed as the mean of the accuracy scores across all folds. The search space is defined as a NumPy array that contains four rows, each corresponding to one of the four hyper-parameters to optimize. A similar approach is used for the stacking classifier, while the final estimator used for the stacking classifier is the voting classifier. It is to be noted that for the problem of classification, the XGBoost model performs or another set of models does not influence the accuracy to a great length, however, to obtain the best possible accuracies, it is best to use the voting classifier as the final estimator. The first column of each row specifies the minimum value for the hyperparameter, and the second column specifies the maximum value. The WOA algorithm is run with the fitness_function and search_space defined above, as well as the number of agents and iterations specified. We used the Nvidia K80 GPU based on cloud for our evaluations on google cloud with 32 GB of usable ram. Three different models are created using the hyperparameters specified in the input parameter X. Limitations of our work include working on an unbalanced and small dataset. Future research should focus on improving the overall effectiveness and security of the framework.

5. Conclusions

TRX is rapidly expanding and has a sizable pool of financial resources. For now, it is not possible to exactly predict how high TRON (TRX) will rise. Based on the price predictions, one can form an opinion about TRON’s future. The volatility of cryptocurrencies makes it difficult to predict significant price variations in the future. However, prediction models have been and will always be useful to predict prices to some extent. It is important to thoroughly examine the ‘market microstructure’ of crypto exchanges. In the past few months, cryptocurrency exchanges have been the focus of some research and have several pressing issues that require further investigation. These approaches may not mean a better assessment of the market directly but may be useful in researching such cryptocurrencies. It is concluded that tree-based regressors combined with staking and voting ensembles techniques using tree-based models in their structure are a good combination for predicting the price and forming an opinion about the future performance of the cryptocurrency based on the historical performance of the tokens. This research is applicable in real-life scenarios and has a wide scope and possesses flexibility in predicting prices in the future as well where the data are limited to make accurate predictions using conventional models.


Funding: This research received no external funding.

Data Availability Statement: The dataset is available on TRONSCAN [46].

Conflicts of Interest: The authors declare no conflict of interest.
Abbreviations

The following abbreviations are used in this manuscript:

AI Artificial Intelligence  
DT Decision Tree  
ET Extra Trees  
GNB Gaussian Naive Bayes  
KNN K-Nearest Neighbors  
LGBM Light Gradient Boosting Machine  
LR Logistic Regression  
MAE Mean Absolute Error  
ML Machine Learning  
MSE Mean Squared Error  
OHL C Open High Low Close  
PCA Principal Component Analysis  
R2 R Squared  
RF Random Forest  
RMSE Root Mean Squared Error  
SGD Stochastic Gradient Descent  
TRX Tronix  
WOA Whale Optimisation Algorithm  
XGB Extreme Gradient Boosting

References

2. Adam, I.O.; Dzang Alhassan, M. Bridging the global digital divide through digital inclusion: The role of ICT access and ICT use. TG 2021, 15, 580–596. [CrossRef]
34. Haykir, O.; Yagli, I. Speculative bubbles and herding in cryptocurrencies. Financ. Innov. 2022, 8, 78. [CrossRef]
45. Li, X.; Liang, C.; Ma, F. Forecasting stock market volatility with a large number of predictors: New evidence from the MS-MIDAS-LASSO model. Ann. Oper. Res. 2022, 1–40. [CrossRef]

50. Shwartz-Ziv, R.; Armon, A. Tabular data: Deep learning is not all you need. *Inf. Fusion* **2022**, *81*, 84–90. [CrossRef]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.