www.editada.org

International Journal of Combinatorial Optimization Problems and Informatics, 15(3), Sep-Dec 2024, 43-58. ISSN: 2007-1558.
https://doi.org/10.61467/2007.1558.2024.v15i3.512

A Comparative Analysis of the Classical and Machine learning Forecasting Methods for the Mexican Stock Exchange

Juan Frausto-Solís[1], Javier Alberto Rangel-González,[2,], Erick Estrada-Patiño[1], Juan Javier González-Barbosa[1], Erika Alarcón Ruiz[1] , Guadalupe Castilla-Valdez[1], Ocotlán Díaz-Parra[3]*

[1]Tecnológico Nacional de México/Instituto Tecnológico de Ciudad Madero, Cuidad Madero, Tamaulipas, México
[2]Universidad Autónoma de Tamaulipas, Morelos, México.
[3]Universidad Politécnica de Pachuca, Pachuca, México

E-mails: juan.frausto@gmail.com, javieralberto64@hotmail.com

Abstract. There is no recent comparison in the literature on the application of classical and machine learning methods for forecasting financial assets on the Mexican Stock Exchange (BMV). These methods divide the time series into three sections: training, validation, and testing. They predict future values using the training data and are evaluated in the validation phase using error metrics; once the lowest error is obtained, the best parameters and algorithms are used to predict values using the test section. This paper aims to find the most accurate regression algorithm to make predictions in the financial time series of the BMV. The regression methods compared include linear regression, neural networks, decision trees, and support vector regression. The study uses historical BMV asset price data to compare the accuracy of each of these algorithms.

Keywords: Statistical forecasting methods, Regression Algorithms, Mexican Stock Exchange, Machine learning forecasting.

Article Info
Received Jan 26, 2024
Accepted July 11, 2024

1 Introduction

Financial markets have been consolidated as elements of great importance for national economies, where companies and traders find business opportunities to increase their capital through stock trading (Roca et al, 2009) (Misra & Chaurasia, 2020). However, those who wish to perform their investment portfolios or purchase and sell assets require unavailable future stock prices; they should estimate them, and forecasting machine learning and statistical methods represent the best alternative to estimate them. The development of advanced technologies and the increased accessibility of financial data have boosted interest in trading-oriented forecasting techniques to reduce the uncertainty inherent in the sector due to fluctuating asset values, market volatility, and macroeconomic and environmental factors (Wasserbacher & Spindler, 2022) (Rouf et al, 2021). Not all assets exhibit the same level of volatility; some maintain specific stability, while others are highly volatile (Idrees et al, 2019). This variability highlights the importance of predicting their future behavior to make informed decisions with high confidence (Granger & Poon, 2024) (Xiao & Aydemir, 2007).

Historically, the behavior of assets has been represented by time series, which capture the asset's value at regular intervals (Weigend, 1994). The graphical representation of the time series allows efficient visualization of the asset's behavior. Furthermore, the numerical data can be used in time series forecasting models to estimate the future values of the stocks, which is of great importance for creating investment portfolios with favorable rates of return (Hyndman & Athanasopoulos, 2021). The most commonly used forecasting models are the classical or statistical models, such as linear regression, exponential smoothing, and autoregressive strategies; besides modern machine learning techniques, such as support vector machines, decision trees, nearest neighbor techniques, and neural networks are common (Dougherty, 2011) (Gerlein et al, 2016) (Makridakis et al, 2008). The later models generally take advantage of the possible data patterns present in the time series, such as the trend, which refers to the behavior of the series in the medium or long term and can be increasing, decreasing, or constant; its seasonality, which refers to constant repetitive patterns within the series; its cyclicality, which refers to very long-term behaviors attributable to

© Editorial Académica Dragón Azteca S. de R.L. de C.V. (EDITADA.ORG), All rights reserved.

macroeconomic factors; and the residual component, which encompasses behaviors over which there is no control (Hyndman & Athanasopoulos, 2021). Understanding these components facilitates understanding the asset's behavior and enables more accurate predictions (Werner, 2016).

The Mexican Stock Exchange (BMV) is one of Mexico's leading financial institutions and is crucial in the domestic financial market. The BMV was established in 1894 to facilitate securities trading. It provides a transparent and efficient environment for purchasing and selling stocks, bonds, and other financial instruments. Its importance lies in channeling national savings into productive investment, boosting the country's economic growth. In addition, the BMV offers a wide range of products and services, including stock market indexes that reflect market performance and advanced tools for financial analysis. As a result, it is an essential point of reference for investors and analysts (Bolsa Mexicana de Valores, n.d.).

In the context of this paper, the selection of assets from the Mexican Stock Exchange enables us to assess the suitability and efficacy of diverse time series forecasting techniques in an emerging and dynamic market (San Martin-Reyna & Duran-Encalada, 2012). The BMV, with its rich data history and relevance in the Mexican economy, provides an ideal scenario to assess forecasting models and gain deeper insights into market dynamics that encompass both stability in specific sectors and high volatility in others (Herrera & Lockwood, 1995). These facts enhance the comparative analysis and provide valuable insights for investors seeking to optimize their trading strategies. This paper presents a comparative study of various time series forecasting techniques applied to predict the value of financial assets. Several assets of the Mexican Stock Exchange (BMV) were selected to analyze each method's capabilities, strengths, and limitations. The objective is to provide an overview of the performance of these methods using this market's data.

This paper is organized as follows: Section 2 introduces the theoretical framework of the related strategies; Section 3 provides a detailed description of the experimental process with the selected strategies; and Section 4 presents the conclusions obtained and a discussion of the work.

2 Related Methods

The following is an overview of the methods and strategies used for comparison in this work. We remark that there are many variants of each method in the literature, each successful for the type of problem addressed. However, it is not possible to establish a definitive superiority for all problems. Therefore, the approaches discussed are described in general terms. Although they have been optimized to produce the best possible results, other variants could improve or worsen the predictive value.

2.1 Time Series

In the forecasting area, following the traditional definition, a time series is defined as a set of one or more variables recorded at regular intervals (Hyndman & Athanasopoulos, 2021). The value that an asset has taken in a specific period of the time series was previously influenced by events that occurred before the said period and whose number of periods and variables affecting it is usually impossible to establish. Thus, all we can say is that the value that has taken an asset in a specific time series has been influenced by the event that occurred in a specific period. In forecasting financial assets, such as those traded on the Mexican Stock Exchange (BMV), these time series permit modeling the evolution of the value of these assets as a function of time (Osisanwo & Atanda, 2012).

In general, each variable in a time series can be decomposed into three main components: the trend, which represents the general direction in which the series moves over the long term; the seasonality, which reflects repetitive patterns over shorter periods; and the residual component, which captures the variability not explained by the trend and seasonality (Zarnowitz & Ozyildirim, 2006). Furthermore, in the case of long-term time series, a cyclical component may emerge that is not associated with seasonality (Harvey & Trimbur, 2003); instead, it may be linked to factors such as economic policies or long-term phenomena (Alexander, 2017).

The objective of a forecasting method is to fit the predicted values to the observed actual values in the given time series. A regression function typically represents the predictions to ensure that the fitted curve is as close as possible to the original curve of the time series; this enables the capture and learning of patterns and behaviors over time with the known data, which in turn makes it possible to produce accurate estimates of the future over a forecast horizon (De Gooijer & Hyndman, 2006). The similarity between regression functions is evaluated using error metrics, which quantify the discrepancy between predicted and actual values (Naser & Alavi, 2023). In the context of this paper, the error metrics used to evaluate the quality of predictions are discussed in the paper.

2.2 Components in time series

Time series are a set of successive observations measured over time, with the characteristic that current observations directly depend on past observations (Araghinejad, 2014). Thus, time series do not present randomized behaviors but follow specific patterns that tend to oscillate over time. These patterns are trend, seasonality, and randomness handled here with residual techniques (Hyndman & Athanasopoulos, 2021). Graphically, we can observe the typical behavior of the components in Figure 1, where we also observe that these components are the product of the differentiation or decomposition of the original time series.

Trend refers to the long-term behavior or movement of the data, i.e., the general direction of the data, which can be increasing, decreasing, or in a direction with a slope close to zero. The trend allows us to understand the general pattern of the data over time (Hamilton, 1994).

Seasonality refers to movements or fluctuations that occur over a fixed period and are repeated within the series (Hyndman & Athanasopoulos, 2021). These variations can increase, decrease, or remain stable, and close prices are commonly used for predicting asset prices. In multivariable forecasting, several variables are considered to predict a single variable (Hamilton, 1994). Nevertheless, in the case of asset price prediction, the Market Efficient Hypothesis establishes that for predicting any asset price, all the information required is contained only in the time series of the asset. Thus, in this work, we only consider this variable (Fama, 1970). The component of the times series called residual is the value that does not adjust to seasonality or trend and is generally a product of the random nature of the data. This component captures irregularities and unforeseen variations that the other two components are unable to explain why these changes happen (Araghinejad, 2014).

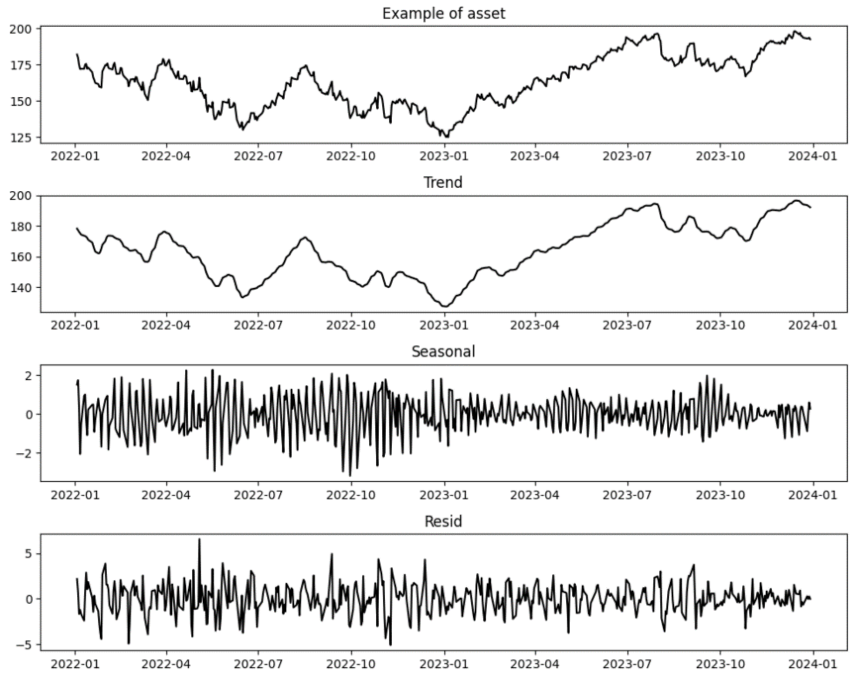

Fig. 1. Example of an asset taken randomly from Yahoo Finance

2.3 Representation of assets as a time series

Financial assets have characteristics that allow their representation through time series, which is applicable not only in the Mexican Stock Exchange (BMV), but in any asset exchange platform. The presentation of these data in a time format is fundamental for their analysis and for applying techniques to forecast their value. A time series is a sequence of values measured in continuous and uniform time intervals; for example, an asset can be represented through its price, trading volume, and derivative indexes in regular intervals such as every hour, every market close, and every week (Rapach & Zhou, 2022).

This temporal representation allows visualizing patterns and trends that are crucial for the development of predictive methods. Asset time series data includes vital information such as price movements, volatility, and cyclical behavior (Idrees et al, 2019). By analyzing these patterns, predictive models can identify trends and make inferences about future market movements. This analysis is essential for making informed investment and risk management decisions (Pincus & Kalman, 2004). In addition, the use of advanced time series analysis techniques and machine learning algorithms, such as autoregressive models, moving averages, recurrent neural networks, and regression methods, enhances the ability to transform historical data into powerful tools for predicting and anticipating the behavior of financial assets (Kumar, 2021).

2.4 Asset Forecasting Models

As mentioned above, there are different approaches to time series forecasting. Depending on the data set used, the methods can have varying performance; no universally superior method exists (Wolpert & Macready, 1997). A brief review of the forecasting methods selected for this work is presented below; these methods were chosen based on the strategies that showed the best average results and are widely studied in the literature.

2.4.1 Linear Regression

Linear regression is one of the most straightforward statistical strategies for modeling the relationship between input and output variables (Altay & Satman, 2005). This method allows an approximation of a linear function that is adjusted to minimize the distance between observed values and model predictions by reducing an error metric (Su et al, 2012). Generally speaking, simple linear regression is expressed by the equation $y = \beta_0 + \beta_1 x + \epsilon$ where *y* is the dependent variable, x is the independent variable, β_0 is the intercept or bias term, β_1 is the regression slope, and ϵ is the error or residual term that captures variations not explained by the model.

One of the main advantages of linear regression is its high interpretability. The β coefficients provide a direct measure of the expected change in the dependent variable per unit change in the independent variable, holding all other variables constant. However, the effectiveness of this model is compromised if the underlying relationships between the variables are not linear or if there is a high presence of outliers, which can lead to an incorrect fit of the data and inaccurate predictions.

2.4.2 Support Vector Regression

Support Vector regression machines (SVR) are a variant of support vector machines (SVM), which are widely used machine learning strategies (Drucker et al, 1996). The model was initially developed by Vapnik, and has been successfully tested in numerous state-of-the-art papers (Cortes & Vapnik, 1995). The SVR strategy generally attempts to fit a kernel to the regression line established by a known data set. As expected, this task does not always result in an exact fit, and it is necessary to determine confidence intervals or boundary areas bounded by an ε value that allows capturing observations outside the regression curve (Balabin & Lomakina, 2011). Values within the interval are accepted, and those outside are considered errors. The idea is to adjust the confidence intervals to small values to reduce the number of values outside the interval, thus reducing the error (Bathla, 2020).

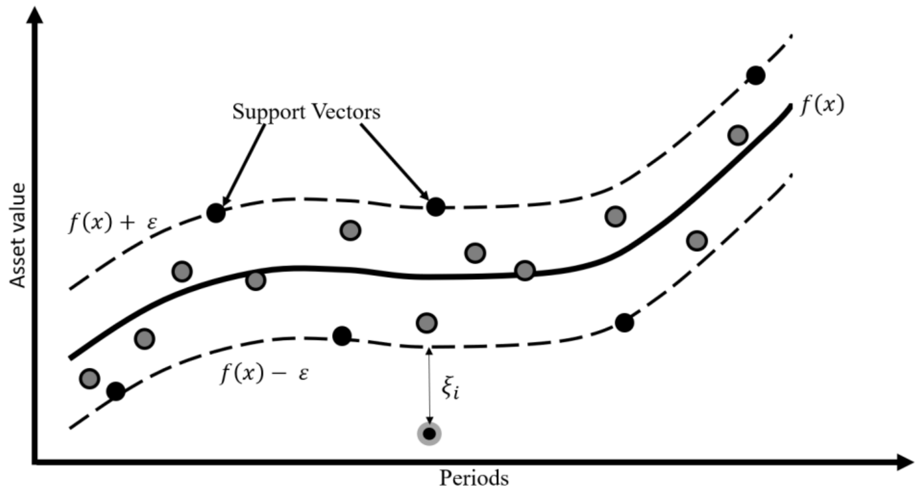

Fig. 2. Adjustment of SVR

In the context of machine learning, a hyperplane is the representation of a multidimensional space, which in SVM is used to separate sets or classes, seeking to maximize the distance between them and thus improve the accuracy of the model. SVR relies on constructing a hyperplane in which all values within a margin ε are accepted and, therefore, are not considered errors, thus allowing to approximate the best value within that margin, as seen in Figure 2.

$$y = \sum_{i=1}^{N} (a_i - a_i^*) \cdot K(x_i, x) + b \tag{1}$$

The standard function representing the prediction of an SVR model is given by equation 1, where y is the model prediction, N is the number of cases in the training set, while $(a_i - a_i^*)$ represent the Lagrange multipliers, which quantify the importance of each data point, looking for a simple and generalizable model; $K(x_i, x)$ are the kernel function where x_i is the feature vector for the *i*-th element, and *x* is the vector to forecast; finally, *b* is the bias value (Drucker et al, 1996).

Within the family of Support Vector Machine algorithms for regression, we find variants such as NuSVR (Nu Support Vector Regression), which incorporates a parameter ν. This parameter controls the fraction of data points used as support vectors, replacing the traditional ϵ value. This strategy allows for an easier-to-interpret model and, to some extent, greater flexibility. However, it can be susceptible to outliers or very noisy data, which could negatively affect the performance (Bathla, 2020).

Linear SVR is another variant of the traditional strategy, designed exclusively to model linear relationships between model variables. It uses a linear hyperplane that minimizes the sum of absolute errors between model predictions and actual output values. This model is usually efficient in terms of processing and interpretation. However, modeling exclusively linear relationships may not be suitable if the variables have nonlinear relationships or high dimensionality, limiting its ability to capture the complexity of the data set (Ho & Lin, 2012).

2.4.3 Decisions trees

Decision trees are a widely used strategy in machine learning, proving effective in classification and regression problems (De Ville, 2013). This model is based on constructing one or more trees that divide a training data set, using the variable that provides the greatest information gain at each step (Ali et al, 2012).

The structure of decision trees is highly interpretable, as it can be visualized in a clear and understandable way. In addition, each tree can be interpreted as a set of decision rules based on the input variables.

Generally, decision trees start with a root node containing the entire data set. The model evaluates all input variables from this node to determine which best divides the data set. This iterative process continues until a predetermined stopping condition is met, which can be a minimum number of samples per node or the maximum depth of the tree. This approach allows the problem to be broken down into more straightforward and manageable decisions, thus facilitating the interpretation and implementation of the model (Apté & Weiss, 1997).

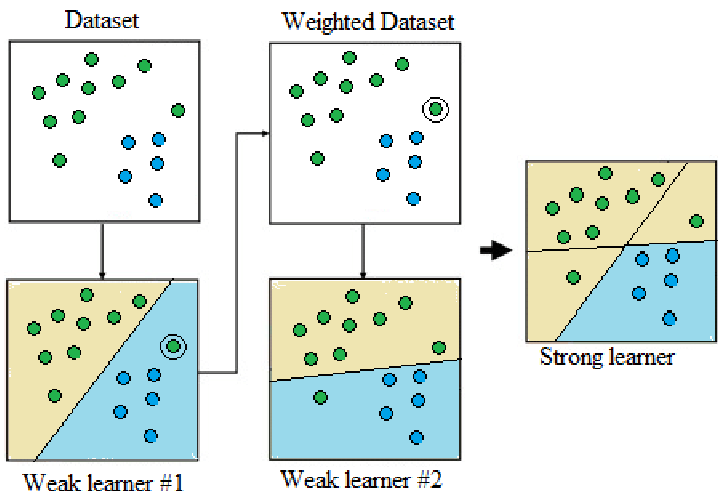

Fig. 3. Principle of AdaBoost algorithm (Imtiaz et al, 2020).

Despite the success of this model, decision trees can be prone to overfitting, mainly when applied to non-representative data sets or when the structures are too complex (Mazumder et al, 2022). To mitigate these problems, it is essential to thoroughly analyze the data and employ pruning techniques that trim unnecessary branches from the tree. In addition, ensemble approaches, such as random forests or boosting, can be used, which combine multiple decision trees to improve the accuracy and robustness of the model. These strategies help to avoid overfitting and ensure that the model generalizes well to new data. In Figure 3, the generic process for AdaBoost is presented. Different strategies, both individual and ensemble or hybridization techniques, are derived from this approach. In this work, the boosting strategies of AdaBoost, an adaptive ensemble technique that improves the efficiency of individual decision trees, are used. AdaBoost applies a sample-weighting approach, focusing on the predictions with the highest error. This approach allows the model to generate a new, improved binary tree, sequentially building trees that correct the mistakes of the previous ones to obtain a better forecast result (Ying et al, 2013).

AdaBoost generally starts by assigning equal weights to all samples in the training dataset. The total number of samples dictates the type of AdaBoost strategy used and how the decision trees are trained; samples that are incorrectly classified are given greater weight in subsequent iterations. In this way, AdaBoost adjusts its approach to focus on the most difficult-to-predict cases, gradually improving the model's accuracy. This iterative process continues until a predefined number of trees is reached or until the prediction error reaches an acceptable threshold. Thus, AdaBoost generates a robust and accurate model and maintains the high interpretability characteristic of decision trees since each tree in the sequence can be visualized as a set of decision rules based on the input variables (Schapire, 2013).

2.4.5 K-nearest neighbors

The k-nearest neighbors (KNN) algorithm is a widely used supervised classification algorithm applied in tasks such as data preprocessing, recommendation engines, finance, healthcare, and pattern recognition (Cover & Hart, 1967). Unlike many machine learning algorithms, KNN does not have an explicit learning phase; instead, it performs its prediction mechanism using the entire available dataset (Guo et al, 2003). Unlike many machine learning algorithms, KNN does not have an explicit learning phase; instead, it performs its prediction mechanism using the entire available dataset (Guo et al, 2003). Its strategy is based on finding similarities between variables in the hyperplane, considering that close values belong to the same neighborhood and, therefore, to the same class or value. Thus, the number of neighbors for searching must be determined, and the neighbors with the shortest distance to the target value should be found. Equations 3, 4, and 5 show the calculation functions for Euclidean, Manhattan, and Minkowski distances, respectively.

$$d(\boldsymbol{p},\boldsymbol{q}) = \sqrt{\sum_{i=1}^{k}(p_i - q_i)^2} \tag{3}$$

$$d(\boldsymbol{p},\boldsymbol{q}) = \sum_{i=1}^{k} |p_i - q_i| \tag{4}$$

$$d(\boldsymbol{p},\boldsymbol{q}) = \left(\sum_{i=1}^{k} (|p_i - q_i|)^p\right)^{1/p} \tag{5}$$

where *d(p,q)* is the distance between the n-dimensional points *p* and *q*, *k* is the number of coordinates, p_i and q_i represent the *i-th* coordinates, and the parameter *p* defines the order of the distance. Once these K neighbors are obtained, the class is chosen through a weighting function. This technique is known for its simple and intuitive implementation. However, predictions can be computationally expensive, especially when the dataset is large. Moreover, KNN can be highly sensitive to outliers and noise, affecting the model's accuracy if the data has not been adequately cleaned (Zhang, 2021).

There are improvements and variants to address some KNN limitations. A common strategy is weighted KNN, which assigns weights to neighbors based on their distance to the data point, giving more importance to closer neighbors. Despite its limitations, KNN is a powerful and versatile algorithm with applications in various fields. Its simplicity and effectiveness, in many cases, make it a valuable tool. Using enhancements and optimizations, KNN can be adapted to handle a wide range of classification and regression problems, providing accurate and valuable results in diverse contexts (Pérez-Ortega et al, 2022) (Pérez-Ortega et al, 2024).

2.4.6 Multilayer Perceptron

The MLP (Multi-Layer Perceptron) regression algorithm is a learning model that employs several hidden layers to process the input data and generate a continuous output (Haykin, 2000) (Stathakis, 2009). The structure of the MLP model consists of multiple layers of interconnected neurons, including an input layer, one or more hidden layers, and an output layer. In the input layer, data are received and distributed to the neurons of the first hidden layer. This initial layer does not perform significant transformations on the data. Still, it gives the foundation for further processing (Popescu, 2009). The hidden layers are crucial in the MLP's ability to learn complex patterns. An activation function is applied on each neuron in these layers, which introduces nonlinearity into the model and allows it to capture more complex relationships (Windeatt, 2006). Among the most common activation functions are the sigmoid function, the hyperbolic tangent (tanh), and the rectified linear unit (ReLU). These functions transform the inputs' weighted sum into an output passed to the next layer. The sigmoid function, for example, maps the inputs to a range between 0 and 1, which is helpful for modeling probabilities. The tanh function maps the inputs to a range [-1 ,1], centering the data and thus facilitating learning in deep networks. The ReLU function, meanwhile, introduces computational simplicity by keeping only positive values and setting negative values to zero, enabling faster and more efficient training of deep models (Rasamoelina et al, 2020).

The MLP output layer produces the final model output, which is a continuous value in the case of regression. Depending on the problem, this layer may contain a single neuron (for single-valued predictions) or multiple neurons (for multivariate predictions). MLP training aims to adjust the weights of all neural connections by minimizing the error between the model predictions and the actual output values. This optimization process is performed using algorithms such as gradient descent, which adjusts the weights based on the error calculated at each iteration. In addition, more advanced variants of gradient descent, such as stochastic gradient descent (SGD) and Adam, can be employed, which improves the efficiency and speed of training.

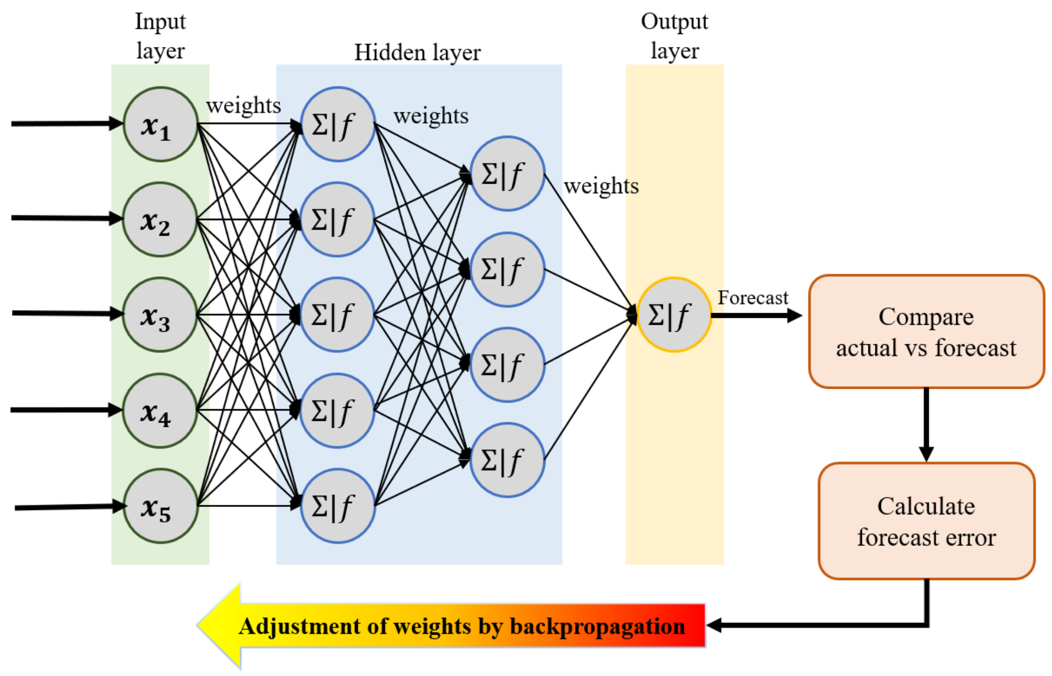

Fig. 4. MLP neural network structure for regression

In Figure 4, we represent how the input data is propagated through the hidden layers and how the activation functions and the adjusted weights influence the generation of the final output. This visualization can also highlight the complete connectivity between neurons in adjacent layers, showing how each neuron in one layer is connected to all neurons in the next layer, enabling feature interdependence during the learning process.

During training, the MLP adjusts the weights through an iterative process of error backpropagation. This process involves calculating the error at the output layer and propagating it backward through the network, adjusting the weights accordingly to reduce the error. Backpropagation is combined with the optimization algorithm to adjust the weights efficiently (Hecht-Nielsen, 1992). This method allows the MLP to learn from the training data and improve its generalization ability to make accurate predictions on new data. The MLP's capabilities to capture complex, nonlinear relationships make it especially useful in many applications, including forecasting, where prediction accuracy is essential. Its flexible structure and the ability to adjust multiple hyperparameters, such as the number of hidden layers, the number of neurons in each layer, and activation functions, allow researchers and practitioners to tailor the model to their specific needs and continuously improve its performance.

2.4.7 Partial least squares

The least squares technique is a statistical strategy widely applied in economics, engineering, data analysis, and many others. Its objective is to find the best approximation of a linear model to a set of observed data (Pirouz, 2006).

$$E(a,b) = \sum_{i=1}^{n} (y_i - (a \cdot x_i + b))^2 \qquad (6)$$

where *E(a,b)* is the sum of squared errors, y_i represents the observed values of the dependent variable, x_i contains the observed values of the independent variable, *a* is the slope, and *b* is the y-intercept of the regression line. This is achieved by minimizing the sum of the squares of the differences between the observed values and the values predicted by the model. Thus, given the set (x_i,y_i), we seek to find the values *a* and *b* on the straight line that minimizes Equation 6 (Cha, 1994). A variant used in this work is partial least squares (PLS), which is used to find linear relationships when the data are highly collinear, finding a data projection that maximizes the covariance between the set of independent and dependent variables. On the other hand, PLS-Canonical seeks to find linear relationships between two sets of multivariate variables, unlike the standard strategy, which maximizes the covariance between two blocks of independent and dependent variables simultaneously (Esposito Vinzi & Russolillo, 2013).

2.4.8 Passive-aggressive algorithm

The Passive-aggressive (PA) algorithm is a powerful machine learning technique designed to minimize model loss by leveraging a unique loss function tailored to the specific learning problem. This approach is particularly effective when the model needs to adapt quickly to new data. When the model makes an accurate prediction, its parameters remain largely unchanged, preserving the knowledge it has already acquired. Conversely, when the model errs, it adjusts its parameters in a direction that compensates for the mistake, thereby refining its accuracy over time. This dual response mechanism—staying passive for correct predictions and becoming aggressive for incorrect ones—enables the PA algorithm to maintain a balance between stability and adaptability (Chang et al, 2010). One of the key strengths of the PA algorithm lies in its simplicity and efficiency. Unlike other machine learning methods that may require extensive computational resources or complex parameter tuning, the PA algorithm operates with straightforward update rules. This makes it particularly suitable for real-time applications where rapid decision-making is crucial. Additionally, the algorithm's ability to handle binary and multi-class classification tasks broadens its applicability across various domains, from spam detection to financial forecasting. By continuously adjusting its parameters in response to errors, the PA algorithm ensures that it remains relevant and accurate, even as the underlying data distribution evolves.

2.4.9 Lasso

Lasso is a regularization technique that effectively selects a subset of essential predictor variables by penalizing the regression coefficients based on their magnitude. By applying this penalty, Lasso encourages the model to shrink some coefficients to zero, thereby excluding less essential variables from the model. This reduction in model complexity helps to avoid overfitting, ensuring that the model generalizes well to new, unseen data. The simplicity and interpretability of the resulting models make Lasso particularly valuable in fields where understanding the underlying relationships between variables is crucial (Ranstam & Cook, 2018). Moreover, Lasso's ability to perform feature selection within the modeling process sets it apart from other regularization methods. Lasso simplifies the model and enhances its predictive performance by automatically identifying and retaining only the most relevant predictors. This is especially beneficial in high-dimensional datasets where the number of predictors exceeds the number of observations. The penalization mechanism not only improves model accuracy but also aids in uncovering the most significant factors influencing the response variable. Consequently, Lasso remains a popular choice for regression analysis, offering a balanced approach to managing model complexity and interpretability (Alhamzawi & Ali, 2018).

3. Experimental Process

The conditions and results of the experimental process are presented below, detailing the obtaining of the instances, the tests with algorithm implementations, the error metrics used and the interpretation of the results.

3.1 Data collection

The assets used in this study belong to the Mexican Stock Exchange (BMV). A sample of 15 assets was taken to evaluate the performance and results of each of the algorithms against the time series of these companies. The selected assets were: ACTINVRB.MX, ALPEKA.MX, ALSEA.MX, AMXL.MX, ASURB.MX, AZTECACPO.MX, FIBRAMQ12.MX, FSHOP13.MX, GBMO.MX, GIGANTE.MX, HOMEX.MX, LABB.MX, PINFRA.MX, POCHTECB.MX and URBI.MX.
The values of these assets were downloaded from Yahoo Finance, covering a 24-month period, from 08/09/2020 to 07/09/2022. In this way, we have 505 data items for each asset in their respective time series. The training phase was performed using the first 485 elements of each series, while the time window for prediction corresponds to the remaining 20 values of each series. This approach allowed the evaluation of the predictive capability of the algorithms on a data set not seen during training, providing a more robust assessment of their performance. The selection of the 15 assets was based on their diversity in terms of industry and market capitalization, aiming to capture a broad spectrum of market behavior. The chosen period from 08/09/2020 to 07/09/2022 included significant market events, such as the recovery phase following the initial impact of the COVID-19 pandemic, which added an extra layer of complexity to the prediction task. Overall, this study contributes to understanding how different predictive models perform in the context of financial time series data, offering valuable insights for investors and analysts in developing more effective forecasting strategies.

3.2 Implementation of the forecasting models

The algorithms used to generate the predictions of each time series were diverse, including machine learning and advanced statistical techniques. These include linear regression, SVR with its variants NuSVR and linear SVR, MLP Regressor, decision tree, and a combination of decision tree with the AdaBoost method using two different configurations for the estimator parameter, with values of 50 and 300. Additionally, Bayesian regression, nearest neighbors, PLS regression, PLS canonical, passive-aggressive regression, and Lasso were included, totaling 15 different algorithms.

The experimentation was conducted using the Python language and the scikit-learn library, which provides a wide range of tools and functions for implementing and evaluating machine learning models. This approach allowed for efficient and reproducible implementation of the algorithms, allowing us to compare their performance in predicting the time series of the selected assets. These algorithms were selected based on their ability to handle time series data and their prevalence in previous financial prediction studies. Multiple experiments were conducted to tune and validate the models, ensuring each was optimized to deliver its best performance. Furthermore, cross-validation techniques were used to assess the models' robustness by splitting the data into multiple subsets to repeatedly train and test the algorithms. This helped mitigate the risk of overfitting and provided a more accurate evaluation of their predictive capabilities. The results obtained from these experiments provide valuable insights into the relative performance of each algorithm but also highlight the importance of selecting the appropriate model based on the specific characteristics of financial data. Therefore, this study offers a valuable resource for researchers and professionals interested in time series prediction in financial markets.

3.3 Error Metric Algorithms in Time Series Forecasting

MAPE measures the prediction errors in percentage, providing an easy interpretation of the model's accuracy.

$$MAPE = \frac{100}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{7}$$

Equation 7 shows the MAPE formula, where y_i represents the actual values, $\hat{y}_i$ the predicted values, and N is the total number of observations. The advantage of MAPE is its ability to be interpreted intuitively, allowing the performance of different models to be compared in relative terms. Moreover, being a standardized metric, it facilitates the comparison of errors between series of different magnitudes. However, MAPE can be influenced by real values close to zero, which can distort its measurement. Despite this limitation, MAPE is widely used in the literature and practical applications due to its simplicity and ease of interpretation.

3.4 Experimentation

The comparison metric between the algorithms was MAPE. In the first phase of the experimentation, 15 assets were selected out of the 76 corresponding to the Mexican Stock Exchange, and the data were cleaned, resulting in a time series of 505 items. The regression algorithms were implemented, and the MAPE was calculated to measure the performance of each algorithm. Tables 1 to 3 show the results of the algorithms, with first and second places shaded in blue and orange, respectively.

Table 1. MAPE results for different regression algorithms

Assets	Lineal Regression	SVR	NuSVR	Linear SVR	MLP
ACTINVRB.MX	**0.00536**	0.0119232	0.0081861	0.1326803	0.0450910
ALPEKA.MX	0.0257636	0.0613773	0.0704218	**0.0185813**	0.0192532
ALSEA.MX	0.2954848	0.0185516	**0.0176108**	0.0198921	0.3018038
AMXL.MX	0.1760139	**0.0236742**	0.0257789	0.0625258	0.2180682
ASURB.MX	**0.0222357**	0.1152942	0.1072539	0.0690743	0.0402737

AZTECACPO.MX	0.3766623	**0.0673360**	0.1709530	0.4937773	0.3411057
FIBRAMQ12.MX	0.0571168	0.0445952	**0.0355279**	0.1928604	0.0502488
FSHOP13.MX	0.0150617	**0.0085153**	0.0087321	0.0969803	0.0643222
GBMO.MX	0.3822901	0.0930643	**0.0607561**	0.7910832	0.3003851
GIGANTE.MX	0.0862386	**0.0446353**	0.0491523	0.2464847	0.0802588
HOMEX.MX	0.4170411	0.6820765	**0.0924926**	0.4358983	0.6844230
LABB.MX	0.1758850	0.1544537	**0.1405102**	0.6620405	0.1864919
PINFRA.MX	**0.0174443**	0.0225572	0.0191010	0.0176238	0.0408576
POCHTECB.MX	0.0888154	0.0177750	0.0157200	0.9597231	**0.0123351**
URBI.MX	0.2033036	0.1774260	0.1618387	0.5173878	**0.0625209**

Table 2. MAPE results for decision trees and Bayesian regression

	Decision Tree	**Decision Tree AB300**	**Decision Tree AB50**	**Decision Tree AB**	**Bayesian Regression**
ACTINVRB.MX	**0.0043461**	0.0047185	0.0047185	0.0056505	0.0053580
ALPEKA.MX	0.0306404	0.0325426	0.0325426	0.0352677	**0.0256860**
ALSEA.MX	0.0373365	0.0453149	**0.0361077**	0.0454955	0.2952312
AMXL.MX	0.0334045	**0.0333927**	0.0334257	0.0344670	0.1759125
ASURB.MX	0.0536812	0.0563681	0.0563681	0.0524089	**0.0221853**
AZTECACPO.MX	0.1132440	0.1086410	**0.1000998**	0.1106137	0.3763246
FIBRAMQ12.MX	0.0355879	0.0330402	**0.0321266**	0.0335353	0.0574206
FSHOP13.MX	**0.0075302**	0.0081633	0.0077545	0.0181335	0.0149623
GBMO.MX	**0.0037096**	0.0053010	0.0053010	0.0058177	0.3817450
GIGANTE.MX	0.0185030	**0.0156415**	**0.0156415**	**0.0156415**	0.0859883
HOMEX.MX	**0.0438634**	0.1104971	0.1104971	0.0739705	0.4170404
LABB.MX	0.0317674	0.0315030	**0.0311961**	0.0312173	0.1753393
PINFRA.MX	0.0204846	0.0200824	0.0200824	0.0217340	**0.0176023**
POCHTECB.MX	**0.0199644**	0.0310731	0.0310731	0.0327981	0.0891546
URBI.MX	**0.0000000**	**0.0000000**	**0.0000000**	**0.0000000**	0.2017392

Table 3. MAPE results for other regression algorithms

	KNN	**PLS Regression**	**PLS Canonical**	**Passive Aggressive**	**Lasso**
ACTINVRB.MX	0.0062669	0.0053634	0.0093543	0.0055041	**0.0051448**
ALPEKA.MX	0.0319678	0.0257636	0.0474303	0.2550770	**0.0253920**
ALSEA.MX	**0.0659569**	0.2954848	0.3523334	0.4080364	0.2951424
AMXL.MX	**0.0324556**	0.1760139	0.1996637	0.1148840	0.1753052
ASURB.MX	0.0536812	0.0222357	0.0390828	**0.0168124**	0.0222203

AZTECACPO.MX	**0.1237246**	0.3766623	0.5035424	0.4782619	0.3611923
FIBRAMQ12.MX	0.0355879	0.0571168	**0.0224625**	0.0552952	0.0576214
FSHOP13.MX	**0.0099524**	0.0150617	0.1076158	0.0321609	0.0130257
GBMO.MX	**0.0035219**	0.3822901	0.4963385	0.1854387	0.3812948
GIGANTE.MX	**0.0160628**	0.0862386	0.1358957	0.2116499	0.0857482
HOMEX.MX	**0.0360502**	0.4170411	1.0165530	1.0000000	0.3229325
LABB.MX	**0.0432522**	0.1758850	0.2604541	0.1309772	0.1751085
PINFRA.MX	0.0193715	**0.0174443**	0.0576212	0.0308392	0.0174646
POCHTECB.MX	**0.0114438**	0.0888154	0.0181548	0.3966782	0.0902525
URBI.MX	**0.0711590**	0.2033036	0.4803909	1.8495231	0.2015713

When executing the algorithms for different time series, we rank the algorithms by the number of times they finish in first and second place; the results are shown in Table 4, where the five best algorithms found can be seen: first, the decision tree with AdaBoost followed by KNN. The performance of the decision tree algorithm with AdaBoost for the assets GBM Corporative (GBMO.MX) and Fibra Shop (FSHOP13.MX) is shown in Figures 5 and 6, respectively.

Table 4. Best Performing Algorithms

Algorithm	Ranked first
Decision Tree AB50	6
KNN	3
PLS Regression	2
SVR	2
Decision Tree	2

The performance of the decision tree algorithm with AdaBoost for the assets GBM Corporative (GBMO.MX) and Fibra Shop (FSHOP13.MX) is shown in Figures 5 and 6, respectively. In Figure 5, we can observe that the algorithm adapts quite well when there is a stationary trend. However, in Figure 6, where volatility is high, the algorithm tends to fail at the steepest peaks, which increases the calculated error.

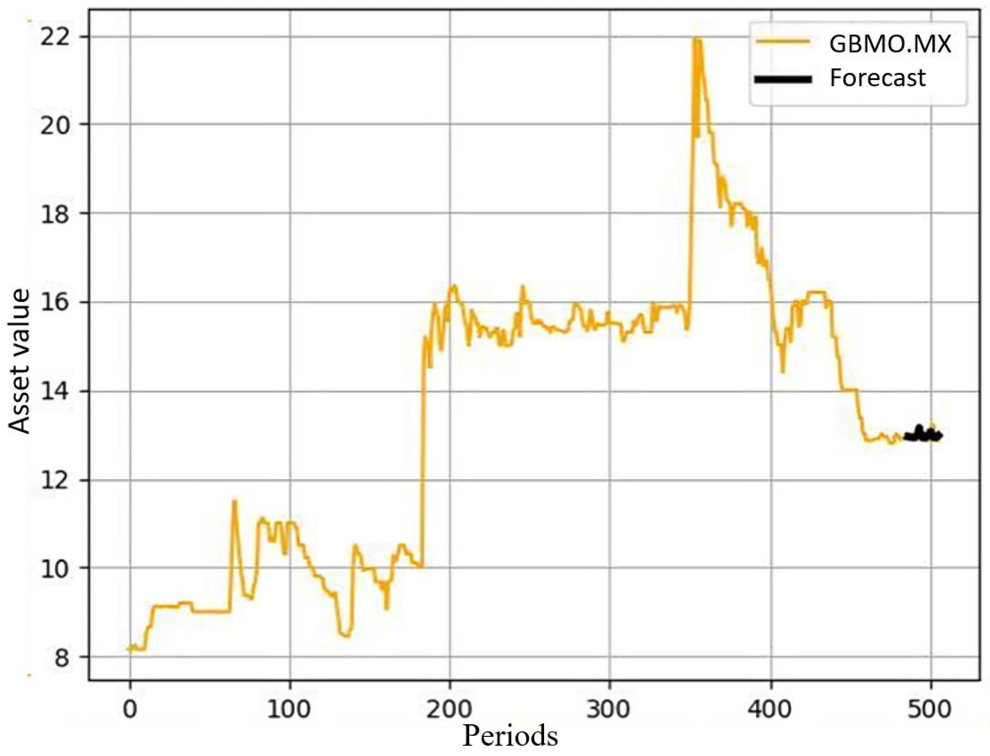

Fig. 5. Performance of GBM Corporative (GBMO.MX)

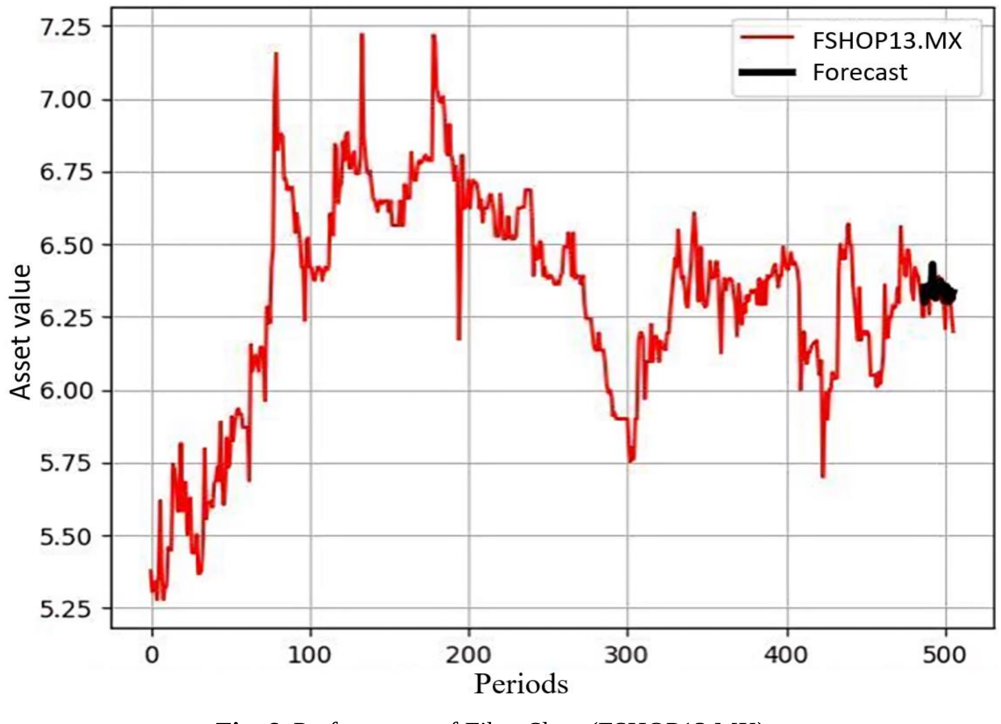

Fig. 6. Performance of Fibra Shop (FSHOP13.MX)

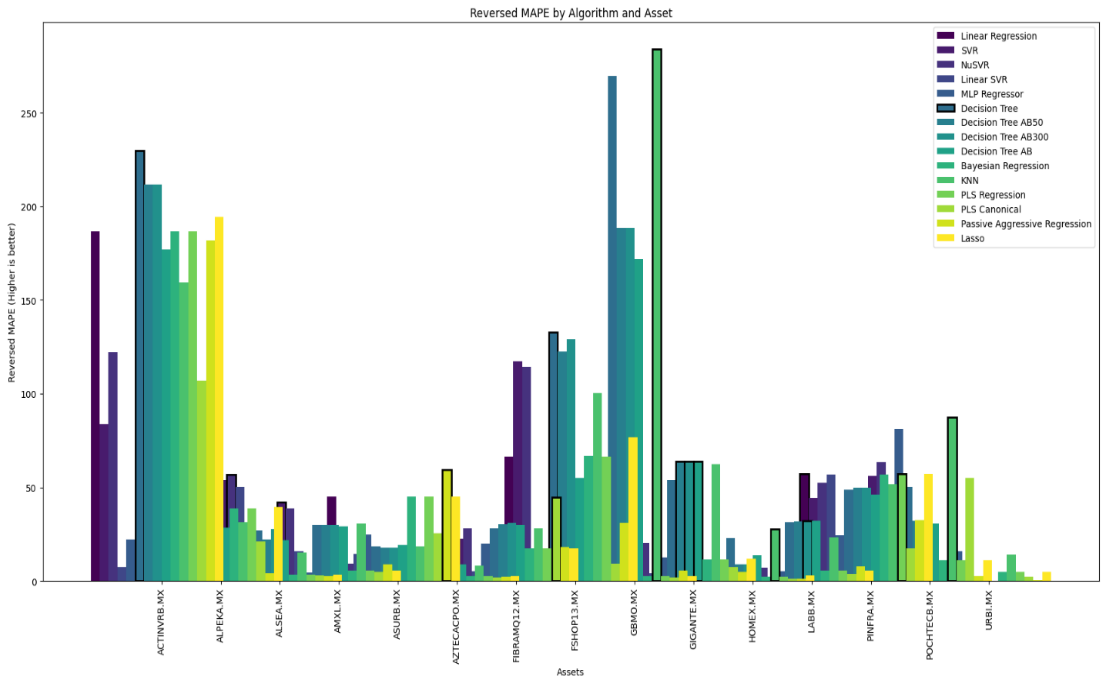

Fig. 7. Reversed MAPE by each algorithm and asset

Figure 7 shows the performance of various machine learning algorithms in terms of the inverse MAPE for different financial assets. The algorithms evaluated include linear regression, support vector machines (SVR), Bayesian regression, decision trees, and partial least squares regression. In the graph, those algorithms with the lowest MAPE for each asset are highlighted with black edges, indicating the most efficient methods for each case. The inversion of the MAPE values allows the most efficient algorithms to be visually highlighted since those with a lower MAPE are shown at the top of the graph. This facilitates the identification of the most accurate methods, providing a clear comparison between algorithms for each asset. Overall, this approach allows a more intuitive and quicker interpretation of the results, highlighting the best options for prediction in the financial domain; with these results, we can conclude that the decision tree algorithm with AdaBoost showed the best performance in terms of MAPE compared to other algorithms, especially in time series with lower volatility.

4 Conclusions

This work compares the classic and machine learning forecasting methods applied to the Mexican Stock Exchange or BMV). The characteristics, advantages, and limitations of each approach were analyzed. While classical forecasting or statistical methods have been used for decades to model and predict time series in financial markets, machine learning methods are more recent and used to capture complex patterns in large volumes of data. The experimentation presented the obtaining of the instances, the tests with the implementations of the algorithms, the error metrics used, and the interpretation of the results. The data sets used correspond to stocks of the Mexican Stock Exchange (BMV) and consist of time series with a temporal format that is essential for their analysis and for applying techniques that allow forecasting their value. Fifteen regression models were selected and implemented to predict the assets of the selected data sets. It was determined which algorithms have a better performance for the set of evaluated instances, highlighting the decision tree with AdaBoost in the first place. In future work, the implementation of an automatic parameterization algorithm using self-generated fuzzy logic will be proposed.

References

Alexander, N. (2017). Factors affecting earnings management in the Indonesian Stock Exchange. *Journal of Finance and Banking Review, 2*(2), 8-14.
Alhamzawi, R., & Ali, H. T. M. (2018). The Bayesian adaptive LASSO regression. *Mathematical Biosciences, 303*, 75-82.
Ali, J., Khan, R., Ahmad, N., & Maqsood, I. (2012). Random forests and decision trees. *International Journal of Computer Science Issues (IJCSI), 9*(5), 272.

Altay, E., & Satman, M. H. (2005). Stock market forecasting: Artificial neural network and linear regression comparison in an emerging market. *Journal of Financial Management and Analysis, 18*(2), 18.

Apté, C., & Weiss, S. (1997). Data mining with decision trees and decision rules. *Future Generation Computer Systems, 13*(2-3), 197-210.

Araghinejad, S. (2014). Time series modeling. En *Data-driven modeling: Using MATLAB® in water resources and environmental engineering* (pp. 85-137). Springer Netherlands. https://doi.org/10.1007/978-94-007-7506-0_4.

Balabin, R. M., & Lomakina, E. I. (2011). Support vector machine regression (SVR/LS-SVM)—an alternative to neural networks (ANN) for analytical chemistry? Comparison of nonlinear methods on near-infrared (NIR) spectroscopy data. *Analyst, 136*(8), 1703-1712. https://doi.org/10.1039/C0AN00387E.

Bathla, G. (2020). Stock price prediction using LSTM and SVR. En *2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC)* (pp. 211-214). IEEE.

Bolsa Mexicana de Valores. (n.d.). Retrieved July 12, 2024, from https://camaradecomercioempresarial.org/wp-content/uploads/wpcfto_files/12a0586a2cbf54b32fd7f21c7bf404f0Material%20de%20apoyo%206%20-%20Johan%20Romero%20Azocar.pdf

Cha, J. (1994). Partial least squares. *Advances in Methods for Marketing Research* (Vol. 407, pp. 52-78).

Chang, C. C., Lee, Y. J., & Pao, H. K. (2010). A passive-aggressive algorithm for semi-supervised learning. En *2010 International Conference on Technologies and Applications of Artificial Intelligence* (pp. 335-341). IEEE.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273-297. https://doi.org/10.1007/BF00994018.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory, 13*(1), 21-27.

De Gooijer, J. G., & Hyndman, R. J. (2006). 25 years of time series forecasting. *International Journal of Forecasting, 22*(3), 443-473.

De Ville, B. (2013). Decision trees. *WIREs Computational Statistics, 5*(6), 448-455. https://doi.org/10.1002/wics.1278.

Dougherty, C. (2011). *Introduction to econometrics*. Oxford University Press, USA.

Drucker, H., Burges, C. J., Kaufman, L., Smola, A., & Vapnik, V. (1996). Support vector regression machines. En *Advances in Neural Information Processing Systems* (Vol. 9). MIT Press.

Esposito Vinzi, V., & Russolillo, G. (2013). Partial least squares algorithms and methods. *WIREs Computational Statistics, 5*(1), 1-19. https://doi.org/10.1002/wics.1239.

Fama, E. F. (1970). Efficient capital markets. *The Journal of Finance, 25*(2), 383-417.

Gerlein, E. A., McGinnity, M., Belatreche, A., & Coleman, S. (2016). Evaluating machine learning classification for financial trading: An empirical approach. *Expert Systems with Applications, 54*, 193-207.

Granger, C. W., & Poon, S. H. (2024). Forecasting financial market volatility: A review. *SSRN*. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=268866.

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. En *On the Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE* (Vol. 2888, pp. 986-997). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-39964-3_62.

Hamilton, J. D. (1994). *Time series analysis*. Princeton University Press. ISBN: 9780691042893.

Harvey, A. C., & Trimbur, T. M. (2003). General model-based filters for extracting cycles and trends in economic time series. *The Review of Economic Statistics, 85*(2), 244-255.

Haykin, S. (2000). Neural networks: A guided tour. *Nonlinear Biomedical Signal Processing, 1*, 53-68.

Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network. En *Neural networks for perception* (pp. 65-93). Academic Press.

Herrera, M. J., & Lockwood, L. J. (1995). The size effect in the Mexican stock market. *Journal of Bank & Finance, 18*(4), 621-632.

Ho, C. H., & Lin, C. J. (2012). Large-scale linear support vector regression. *Journal of Machine Learning Research, 13*(1), 3323-3348.

Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: Principles and practice*. OTexts.

Idrees, S. M., Alam, M. A., & Agarwal, P. (2019). A prediction approach for stock market volatility based on time series data. *IEEE Access, 7*, 17287-17298.

Imtiaz Khan, N., Mahmud, T., Islam, M. N., & Mustafina, S. N. (2020). Prediction of cesarean childbirth using ensemble machine learning methods. https://doi.org/10.1145/3428757.3429138.

Kumar Dubey, A., Kumar, A., García-Díaz, V., Kumar Sharma, A., & Kanhaiya, K. (2021). Study and analysis of SARIMA and LSTM in forecasting time series data. *Sustainable Energy Technologies and Assessment, 47*, Article 101474. https://doi.org/10.1016/j.seta.2021.101474

Makridakis, S., Wheelwright, S. C., & Hyndman, R. J. (2008). *Forecasting methods and applications*. John Wiley & Sons.

Mazumder, R., Meng, X., & Wang, H. (2022). Quant-BnB: A scalable branch-and-bound method for optimal decision trees with continuous features. En *International Conference on Machine Learning* (pp. 15255-15277). PMLR.
Misra, P., & Chaurasia, S. (2020). Data-driven trend forecasting in stock market using machine learning techniques. *Journal of Information Technology Research (JITR)*, *13*(1), 130-149.
Naser, M. Z., & Alavi, A. H. (2023). Error metrics and performance fitness indicators for artificial intelligence and machine learning in engineering and sciences. *Applications of Artificial Intelligence and Machine Learning in Architecture, Structure, and Construction*, 3(4), 499-517. https://doi.org/10.1007/s44150-021-00015-8.
Osisanwo, B. G., & Atanda, A. A. (2012). Determinants of stock market returns in Nigeria: A time series analysis. *African Journal of Scientific Research, 9*(1).
Pérez-Ortega, J., Moreno-Calderón, C. F., Roblero-Aguilar, S. S., Almanza-Ortega, N. N., Frausto-Solís, J., Pazos-Rangel, R., & Rodríguez-Lelis, J. M. (2024). A new criterion for improving convergence of fuzzy C-means clustering. *Axioms, 13*(1), 35. https://doi.org/10.3390/axioms13010035.
Pérez-Ortega, J., Roblero-Aguilar, S. S., Almanza-Ortega, N. N., Frausto-Solís, J., Zavala-Díaz, C., Hernández, Y., & Landero-Nájera, V. (2022). Hybrid fuzzy C-means clustering algorithm oriented to big data realms. *Axioms*, *11*(8), 377. https://doi.org/10.3390/axioms11080377.
Pincus, S., & Kalman, R. E. (2004). Irregularity, volatility, risk, and financial market time series. *PNAS, 101*(38), 13709-13714. https://doi.org/10.1073/pnas.0405168101.
Pirouz, D. M. (2006). An overview of partial least squares. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.1631359.
Popescu, M. C., Balas, V. E., Perescu-Popescu, L., & Mastorakis, N. (2009). Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, *8*(7), 579-588.
Ranstam, J., & Cook, J. A. (2018). LASSO regression. *Journal of the British Surgery, 105*(10), 1348.
Rapach, D., & Zhou, G. (2022). Asset pricing: Time-series predictability. *Oxford Research Encyclopedia of Economics and Finance*. https://doi.org/10.2139/ssrn.3941499.
Rasamoelina, A. D., Adjailia, F., & Sinčák, P. (2020). A review of activation function for artificial neural networks. En *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)* (pp. 281-286). IEEE.
Roca, J. C., García, J. J., & De La Vega, J. J. (2009). The importance of perceived trust, security and privacy in online trading systems. *Information Management & Computer Security, 17*(2), 96-113.
Rouf, N., Malik, M. B., Arif, T., Sharma, S., Singh, S., Aich, S., & Kim, H. C. (2021). Stock market prediction using machine learning techniques: A decade survey on methodologies, recent developments, and future directions. *Electronics, 10*(21), 2717. https://doi.org/10.3390/electronics10212717.
San Martin-Reyna, J. M., & Duran-Encalada, J. A. (2012). The relationship among family business, corporate governance and firm performance: Evidence from the Mexican stock exchange. *Journal of Family Business Strategy, 3*(2), 106-117.
Schapire, R. E. (2013). Explaining AdaBoost. En *Empirical Inference* (pp. 37-52). Springer. https://doi.org/10.1007/978-3-642-41136-6_5.
Stathakis, D. (2009). How many hidden layers and nodes? *International Journal of Remote Sensing*, *30*(8), 2133-2147. https://doi.org/10.1080/01431160802549278.
Su, X., Yan, X., & Tsai, C. (2012). Linear regression. *WIREs Computational Statistics, 4*(3), 275-294. https://doi.org/10.1002/wics.1198.
Wasserbacher, H., & Spindler, M. (2022). Machine learning for financial forecasting, planning and analysis: Recent developments and pitfalls. *Digital Finance, 4*(1), 63-88. https://doi.org/10.1007/s42521-021-00046-2.
Weigend, A. S. (1994). *Time series prediction: Forecasting the future and understanding the past* (1ra ed.). Routledge. https://doi.org/10.4324/9780429492648.
Werner, R. A. (2016). A lost century in economics: Three theories of banking and the conclusive evidence. *International Review of Financial Analysis, 46*, 361-379. https://doi.org/10.1016/j.irfa.2015.08.014.
Windeatt, T. (2006). Accuracy/diversity and ensemble MLP classifier design. *IEEE Transactions on Neural Networks, 17*(5), 1194-1211.
Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67-82.
Xiao, L., & Aydemir, A. (2007). Volatility modelling and forecasting in finance. En *Quantitative finance: Forecasting volatility in the financial markets* (3ra ed.). https://doi.org/10.1016/B978-075066942-9.50003-0.
Ying, C., Qi-Guang, M., Jia-Chen, L., & Lin, G. (2013). Advance and prospects of AdaBoost algorithm. *Acta Automatica Sinica, 39*(6), 745-758.
Zarnowitz, V., & Ozyildirim, A. (2006). Time series decomposition and measurement of business cycles, trends and growth cycles. *Journal of Monetary Economics, 53*(7), 1717-1739.
Zhang, S. (2021). Challenges in KNN classification. *IEEE Transactions on Knowledge and Data Engineering, 34*(10), 4663-4675.