Received 6 November 2024
Revised 10 February 2025
9 April 2025
Accepted 16 April 2025

Valuation of American options using machine learning: beyond Longstaff–Schwartz and hybrid models

Maria Vivas-Redondo, María Coronado-Vaca and Esther Vaquero-Lafuente
Facultad de Ciencias Economicas y Empresariales, Universidad Pontificia Comillas, Madrid, Spain

Abstract

This work explores the potential of different machine learning (ML) algorithms in the valuation of American options (Aos), contrasting them with the Longstaff–Schwartz (L–S) model. To carry out this research, the algorithms K-Nearest Neighbors (KNN), Random Forest (RF), Multi-Layer Perceptron (MLP) and Convolutional Neural Network (CNN) are employed using RStudio. The project specifically targets the prediction of the price of Apple's put Aos through regression, utilizing data extracted from Bloomberg as a case study. To evaluate the model's performance in a multidimensional context, we use both historical and stochastic volatility. The results show that these ML algorithms achieve a notable improvement in the performance and accuracy of Aos price predictions compared to the L-S model. RMSE values are very similar using historical and stochastic volatility, the most notable difference appearing in the L-S model. Prior trends in the literature show the development of hybrid models, which combine traditional techniques with the predictive capabilities of ML algorithms in the valuation of Aos in a more efficient and accurate way than the L-S model. However, our paper determines whether the supervised training of ML algorithms exclusively with historical financial market data, without relying on traditional methods, achieves better results than those of the L-S model. This ML "stand-alone" approach to price Aos faces the inability to derive hedging strategies and optimal exercise conditions. Future efforts could explore integrating deep reinforcement learning to identify optimal exercise policies or developing hybrid models that combine ML with traditional frameworks.

Keywords American options valuation, Machine learning, Longstaff-Schwartz, Predictive analytics, KNN (K-Nearest Neighbors), RF (Random Forest), MLP (Multi-Layer Perceptron), CNN (Convolutional Neural Network)

Paper type Research paper

1. Introduction

American options (Aos) pricing represents a significant challenge within the financial industry and has been a topic widely analyzed (Cox *et al*., 1979; White and Hull, 1990; Longstaff and Schwartz, 2001). Black-Scholes (B-S) model (1973), designed to value European options, cannot fully capture the flexibility of exercising Aos before maturity. Since the B-S model assumes the option can only be exercised at maturity, there is a risk of underestimation of its intrinsic value. Therefore, more sophisticated models should be used to avoid non-optimal investment decisions and inefficient financial risk management.

Within this framework, some authors emphasize the importance of determining the expected continuation value in every single moment to optimize the exercise strategy and maximize potential returns. The Longstaff-Schwartz (L-S) (2001) model introduced a

Journal of Derivatives and Quantitative Studies: 선물연구
Vol. 33 No. 3, 2025
pp. 170-188
Emerald Publishing Limited
e-ISSN: 2713-6647
p-ISSN: 1229-988X
DOI 10.1108/JDQS-11-2024-0044

numerical approach based on Monte Carlo simulations and least squares regression to estimate the continuation value of an option. Within the financial industry, this predominant approach is also known as Least Square Monte Carlo (LSMC). The L-S model provides an effective methodology for valuing Aos, yet it still relies on certain assumptions that may limit its accuracy in highly volatile or non-normal market conditions.

Consequently, traditional methods exhibit notable limitations in handling the complexity of American option valuation, thus creating a significant opportunity for ML algorithms in determining their pricing. ML offers a data-driven approach that captures complex, non-linear relationships from historical financial data, eliminating the need for rigid mathematical models. This capability allows ML-based methods to dynamically adapt to changing market conditions, enabling a more precise and flexible valuation. Unlike traditional models, which rely on pre-defined assumptions about asset price behavior, ML algorithms can learn directly from large datasets, identifying hidden patterns and improving predictive performance.

In recent years, the new technologies and ML have revolutionized the valuation of Aos by introducing computational tools that enhance modeling and forecasting capabilities. ML approaches facilitate processing vast amounts of market data from platforms like Bloomberg and Reuters (Arévalo De Pablos *et al*., 2024), ensuring real-time access to reliable and up-to-date financial information. This is particularly crucial for Aos, whose valuation is highly sensitive to market fluctuations and optimal exercise strategies.

Furthermore, ML models can incorporate a broader range of financial variables than traditional methods, leading to more robust and adaptive pricing frameworks. This allows for enhanced valuation accuracy and improved risk management and investment decision-making.

Recently, various authors have used ML to value Aos. Several studies have combined ML algorithms with traditional techniques, such as PDE and LSMC, to value Aos. Among others, Anderson and Ulrych (2023) as well as Becker *et al.* (2020) used Deep Neuronal Networks (DNN); Hoshisashi and Yamada (2023) applied Multilayer Perceptron (MLP); Kanashiro Felizardo *et al*. (2022) utilized Convolutional Neural networks (CNN); Dubrov (2015) and Maidoumi *et al.* (2023) used random forest (RF); Feng *et al.* (2013) preferred KNN; meanwhile Malpica and Frias (2019) applied RF, K-Nearest Neighbors (KNN), Light Gradient-Boosting Machine (LGBM) and a combination of these algorithms through the stacking technique.

These works show the diversity of approaches and the trend towards hybrid models, which combine traditional techniques with the predictive capabilities of ML algorithms for Aos valuation. The conclusions of those research studies are summarized in Table 1.

These conclusions (Table 1) show that using hybrid models to identify the conditional expected continuation value and the optimal stopping rule in Aos is more efficient and accurate than the L-S model. However, it still has to be determined whether the supervised training of ML algorithms exclusively with historical financial market data, without relying on traditional methods, achieves better results than the L-S model.

Therefore, this work aims to develop an approach to predict the price of Aos using only ML algorithms and historical financial market data. The aim is to avoid hybrid models, such as those implemented by the aforementioned authors, and focus on training the KNN, RF, MLP, and CNN algorithms using the most influential financial market factors in Aos. This approach increases adaptability and simplifies the valuation process, avoiding the complexity associated with the mathematical formulas used in hybrid models. The hypothesis is that ML has the potential to value Aos more precisely and efficiently than the L-S model. In order to validate it, different algorithms are applied to a specific case—Apple American put options—in RStudio, comparing them to each other and to the L-S model.

This paper is structured as follows. Section 2 presents an empirical analysis of the case, including data acquisition, preprocessing, and a summary of the methodology. It also provides a step-by-step explanation of the ML algorithms and L-S model used to value Apple American put options in RStudio. Section 3 discusses the results and conclusions.

Table 1. Hybrid models for estimating the price of American options

Author/Year	Hybrid ML model	Conclusions
Anderson and Ulrych (2023)	DNN	• Training DNN with PDE and the Heston stochastic volatility model (1993) improves price prediction in terms of the balance between speed and accuracy, compared to standard methods for valuing American options
Becker *et al.* (2020)	DNN	• The adaptation of LSMC using DNN provides an estimation of the American option price with low bias • Accurate training of DNN requires more computational time
Dubrov (2015)	RF	• RF trained with LSMC always achieves better results than the L-S model • Its simplicity and accuracy make it an ideal algorithm
Feng *et al*. (2013)	KNN	• The Root Mean Square Error (RMSE) of the numerical experiments shows that KNN trained with LSMC is promising for pricing American options • Further research is needed to determine the dimension threshold required for KNN estimators to be viable
Hoshisashi and Yamada (2023)	MLP	• Training MLP with LSMC is not much better than the L-S model when there are few exercise dates. However, its efficiency and accuracy in terms of RMSE are better when there are many dates • High efforts and computational resources are required • It is necessary to frequently train MLP in response to financial market conditions
Kanashiro Felizardo *et al*. (2022)	CNN	• Training CNN with LSMC allows for improving the optimal stopping point performance • The improvement is achieved by transforming historical information into a Markov state, along with extracting features from the CNN layer • The results show that this methodology improves the expected value compared to the L-S model
Malpica and Frias (2019)	RF, KNN, LGBM and Stacking	• The RF, KNN, and LGBM models trained with LSMC achieve approximate accuracy in terms of Mean Absolute Error (MAE) with no significant differences • Their combination through stacking increases accuracy in terms of MAE • On average, the price estimates approach their real value
Maidoumi *et al.* (2023)	RF	• The price estimated by RF trained with LSMC is similar to that obtained through the L-S model, but slightly higher in terms of Mean Squared Error (MSE) • RF generally performs better in the context of nonlinear, highly correlated multidimensional models due to its random tree structure

Source(s): Table by authors

2. Material and methods

2.1 Data

2.1.1 Data sources. The database on which this work is based contains a daily historical record with information on various Aos of Apple, covering the period from January 18, 2018, to January 19, 2024. The data has been collected from Bloomberg.

During the analysis, the following challenge arose: while Bloomberg terminal's "OMON" provides information on active options, it does not include data on expired options. Since it was necessary to extract historical data on expired options, the following Excel's function has been used "= BDS('AAPL US Equity'; 'OPT_CHAIN'; 'SINGLE_DATE_

OVERRIDE = YYYYMMDD')", which returns the tickers of all the options within the specified time range. Subsequently, the historical data of Apple's Aos, as well as Apple's stock and US Treasury bonds, has been extracted using the historical data table from Bloomberg's Excel Add-In.

Due to the diversity of formats and the large volume of information, macros were created in Excel Visual Basic to standardize the data and adapt it for implementation in RStudio. The data on Aos has been divided into two Excel files, "Appleoptions1" and "Appleoptions2", due to Excel's row limitations. Observations on Apple stock and US Treasury bonds have been saved in two other Excel files, "Applestocks" and "TNOTES".

After importing the Excel files into RStudio, a complete dataset was built by merging them using "DATES" as the common element. As a result, 1,661,772 observations and 23 variables were obtained, totaling approximately 0.3 gigabytes of memory.

2.1.2 Data pre-processing. In the dataset, there are missing values, and different alternatives have been considered for handling them, such as taking the mean, maximum, minimum, zero, or deleting all data for the corresponding option on that day. Ultimately, the latter alternative was chosen, as missing data spanned several consecutive days, which could significantly impact the results due to the high volatility in Aos. This reduced the number of observations to 988,384.

Additionally, the dataset contained information on both American call and put options, so the American put options to be used for analysis were extracted. As a result, the dataset is reduced to 476,395 observations corresponding to 1,267 American put options.

To facilitate data handling, a random sample of 5% of the total observations (23,819) was selected for efficient analysis and model building, ensuring minimal impact on accuracy.

Besides, the data was split into training and testing sets. In the training set (70%, 16,673 observations), the model was fitted and estimated to learn from the data and identify the relationships. Meanwhile, the testing set (30%, 7,146 observations), was reserved for making predictions and comparing the predicted values of the target variable with the actual values, allowing for the evaluation of prediction error and predictive performance measures.

To avoid biases, the data was assigned to the training and testing sets randomly, ensuring proper representation in both sets for accurate evaluation of the model.

2.1.3 Variables. From the 23 variables extracted from Bloomberg, four new ones have been created that are considered relevant for predicting the price of American put options: "DAYS_UNTIL_MATURITY", "PAYOFF", "MAX_PAYOFF_PER_OPTION" y "MAXIMUM_DATE_PAYOFF_OPTION".

First, "DAYS_UNTIL_MATURITY" indicates the remaining days until the expiration of each option. The proximity to expiration is a crucial factor in determining the price. As expiration approaches, uncertainty about the future behavior of Apple's stock decreases, which can influence volatility and, therefore, the options price. Additionally, the fewer days remaining, the lower the time value of the options.

Second, "PAYOFF" is the difference between the current price of Apple's stock and the strike price of the Apple American put option. A positive payoff implies profits, while a negative payoff results in losses. Knowing the payoff is key to deciding whether to exercise the option.

Third, "MAX_PAYOFF_PER_OPTION" records the historical maximum payoff for each option. In this study, since a random sample of daily observations from various Apple American put options is being used, this variable helps identify patterns or trends in price behavior and provides a reference for evaluating their future performance.

Finally, "MAXIMUM_DATE_PAYOFF_OPTION" indicates how many days remain until the expiration of each option when the maximum payoff occurs. Understanding at what point in the life of the option the highest historical payoff was achieved can provide insights into how proximity to expiration may influence profitability.

Additionally, since Bloomberg provides the annual volatility of the stock but not the daily volatility, the latter has been calculated using Bloomberg annual volatility. This variable is

more relevant when working with daily historical data, as it offers a clearer view of daily price fluctuations and a more effective risk assessment and management.

To evaluate the performance of ML predictions in a multidimensional context, the analysis will be conducted twice: once using historical volatility and once using stochastic volatility. The stochastic volatility ("VOLATILITY_ST") has been estimated with a GARCH(1,1) model without ARMA terms in the mean equation. The model has been fitted to the return series using the "Rugarch" package in RStudio, and the conditional daily volatilities have been extracted using the sigma function.

As a result, the total number of variables amounts to 29, including qualitative, quantitative, and temporal variables.

To simplify the analysis and avoid multicollinearity, categorical, temporal, and highly correlated numerical variables deemed unnecessary have been removed. A correlation matrix (Figure 1) was used to assess the relationships between variables and to identify those that are highly correlated. Specifically, the target variable "PX_LAST_OPTION" has been excluded.

Based on the correlation matrix analysis, the following variables have been removed: "VOLATILITY_360D", "VOLATILITY_30D", "DIVIDEND", "DELTA_LAST", "PAYOFF", "USGOVT3MONTHS_PX_LAST", "USGOVT6MONTHS_PX_LAST", "USGOVT2YEARS_PX_LAST" and "USGOVT5YEARS_PX_LAST". These variables have a high correlation with others and are redundant in the analysis. Regarding US Treasury bonds, which are used as an approximation for the risk-free rate, those with an annual interest payment frequency have been selected, as the analysis focuses on Apple American put options with an annual expiration.

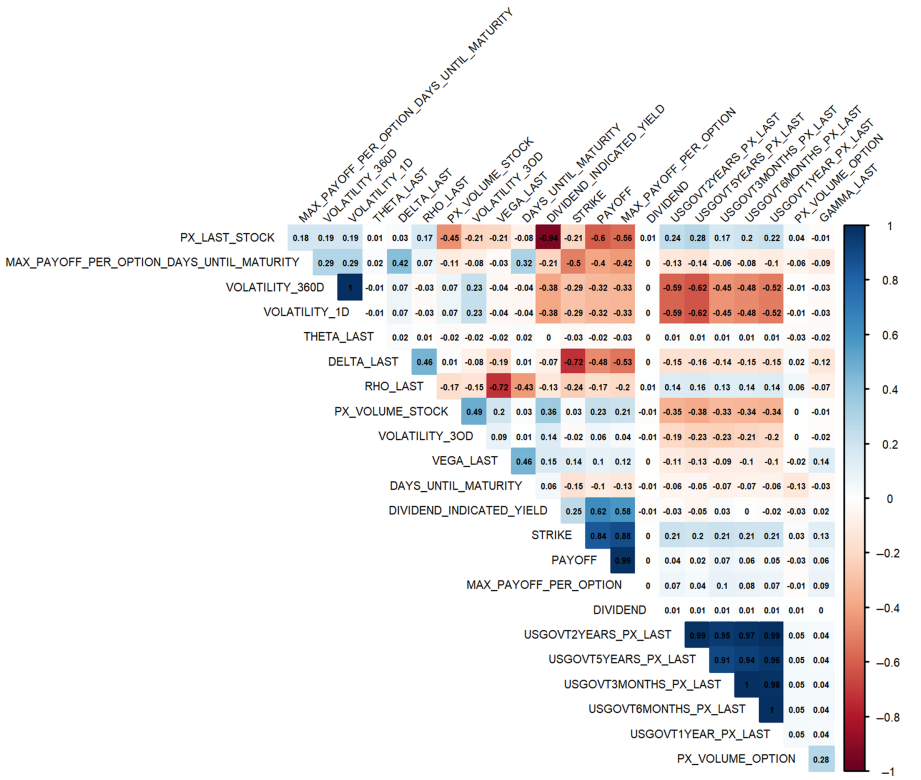


Figure 1. Correlation matrix. Figure by authors

Furthermore, the variables “GAMMA_LAST”, “VEGA_LAST”, “RHO_LAST”, “THETA_LAST” and “PX_VOLUME_OPTION” which inherently rely on existing models (Black-Scholes) or observed outcomes (directly correlate with PX_LAST_OPTION), were excluded to ensure the independence of the proposed model and enhance its ability to predict American put options prices autonomously, free from the assumptions embedded in traditional models. This adjustment aligns the study with its primary objectives, emphasizing innovation and strengthening the predictive validity of the ML-based approach.

Although some variables like “STRIKE”, “DIVIDEND_INDICATED_YIELD”, and “PX_LAST_STOCK”, show high correlation, it has been deemed necessary to include them in the analysis to properly assess Apple American put options. Moreover, all these variables are required to apply the L-S model later.

As a result, the numerical variables are reduced to 11 (Table 2). The ML models will be executed twice: once using historical volatility and once using stochastic volatility, allowing for a comparative analysis of their outcomes. Consequently, 9 variables are predictive, and “PX_LAST_OPTION” is the target variable.

It is important to highlight that, in the context of the L-S model, the following variables have been specifically selected: “STRIKE”, “VOLATILITY_1D”, “PX_LAST_STOCK”, “USGOVT1YEAR_PX_LAST”, “DIVIDEND_INDICATED_YIELD”, and “YEARS_UNTIL_MATURITY”. The variable “YEARS_UNTIL_MATURITY” was created from the “DAYS_UNTIL_MATURITY” variable and shows the remaining time to expiration in annualized terms. The need for this variable lies in its inclusion in the L-S model, where it is used to calculate the expected present value of the American put option over time.

2.2 Methodology

In this step, the ML algorithms -KNN, RF, MLP, and CNN- selected to solve the regression problem and the L-S model will be explained, implemented and analyzed.

Table 2. List of variables included in ML algorithms with variable type and description

Variable	Type	Description
STRIKE	Continuous quantitative	Daily strike price of American put options
PX_LAST_OPTION	Continuous quantitative	Daily price of American put options
PX_LAST_STOCK	Continuous quantitative	Daily stock price
PX_VOLUME_STOCK	Discrete quantitativ5e	Daily stock transaction volume
DIVIDEND_INDICATED_YIELD	Continuous quantitative	Indicated dividend yield of a stock
USGOVT1YEAR_PX_LAST	Continuous quantitative	Daily price of US Treasury bonds with annual maturity, used as an approximation of the risk-free rate
DAYS_UNTIL_MATURITY	Discrete quantitative	Days remaining until the expiration of American put options
MAX_PAYOFF_PER_OPTION	Continuous quantitative	Maximum profitability achieved by American put options
MAX_PAYOFF_PER_OPTION_ DAYS_UNTIL_MATURITY	Discrete quantitative	Days remaining until the expiration of American put options when maximum profitability was achieved
VOLATILITY_1D	Continuous quantitative	Daily stock volatility
VOLATILITY_ST	Continuous quantitative	Daily stock stochastic volatility

Source(s): Table by authors

2.2.1 KNN. The KNN algorithm is a widely used Supervised Learning technique in both classification and regression problems. KNN is a non-parametric algorithm, meaning it does not require a specific training process. Since KNN does not have a specific learning stage, it is a lazy learning algorithm. This algorithm determines the outcome by calculating the distance between the new sample and the existing samples in the training set (Klidbary and Arabameri, 2023). In other words, upon receiving a new instance X, the algorithm searches the entire training set for the K cases most similar to it (the K nearest neighbors). Depending on the problem, the neighbors are used to either classify the sample into a specific category (classification) or to predict its summary value by averaging the K nearest neighbors' values (regression).

The appropriate choice of the hyperparameter K is crucial, as a very small value can lead to overfitting by adjusting too much to the noise in the data, while a value that is too large can introduce excessive bias into the model. Therefore, it is important to find a balance to achieve accurate and generalizable results. The simplicity, easy implementation and interpretability of the results, high accuracy, suitability for non-linear data, and a wide range of applications can be considered as the advantages of the KNN algorithm (Klidbary and Arabameri, 2023).

Next, the KNN algorithm is applied to Apple's American put options data to predict their price. The KNN model has been built using the "train()" function from the "Caret" package in RStudio. For training, the normalized data from the training set was used so that all variables were on the same scale. "PX_LAST_OPTION" is the target variable, and all other variables are used for prediction. The method argument specifies the use of KNN, and 10-fold cross-validation is employed to evaluate model performance—training on 9 folds and testing on the 10th, repeated 10 times for reliability. For hyperparameter selection, "tuneLength" has been used, specifying 15 iterations to find the optimal K.

In Figure 2, it can be observed how, depending on the value of K, the model will have a different RMSE value. RMSE is a metric that provides the average difference between the predicted and actual values. The optimal number of K is five, as it minimizes the RMSE value.

Since the data is normalized, it is returned to its original state to compare the KNN model's price predictions with the original data of Apple's American put options.

In the histogram (Figure 3), it can be observed that most of the errors are concentrated around 0, and there are minimal errors greater than 50 in absolute terms. The highest frequency corresponds to negative values close to 0. This suggests that, overall, the KNN model tends to

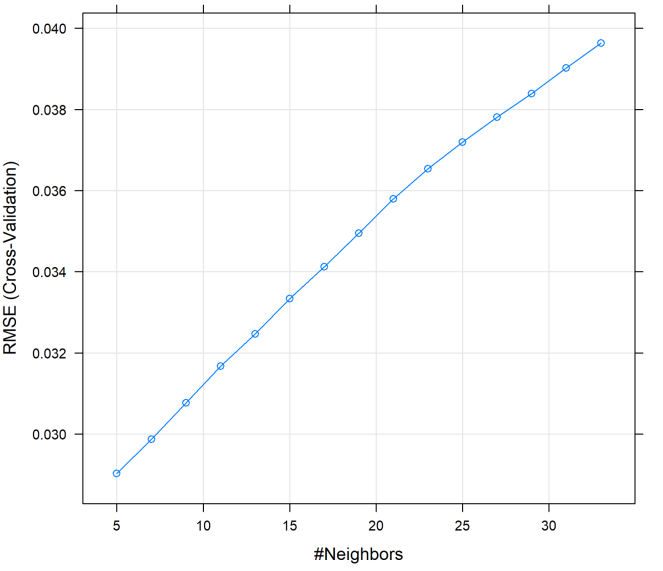


Figure 2. RMSE graph as a function of K. Figure by authors

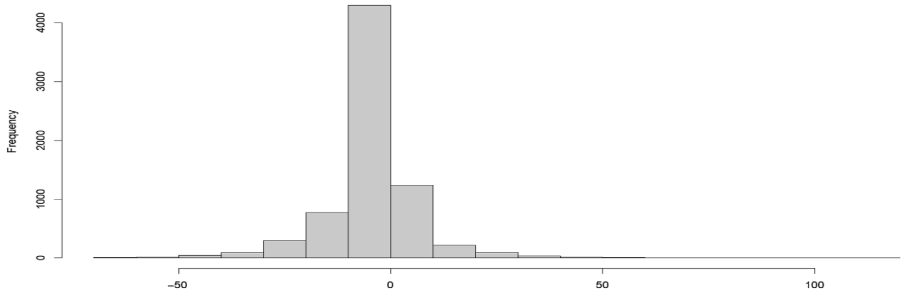


Figure 3. Histogram of KNN prediction errors. Figure by authors

produce results close to the expected value and performs well. However, the fact that there are more negative errors indicates a tendency to overvalue Apple's American put options.

In order to assess the accuracy of the KNN model, its RMSE was calculated, yielding a value of 11.28885. This means that, on average, the model's predictions differ by approximately 11.29 units from the actual values.

Finally, when using stochastic volatility instead of historical volatility to evaluate prediction performance in a multidimensional context, the RMSE is 11.40038. This indicates that the model performs very similarly, having the choice minimal impact on prediction accuracy.

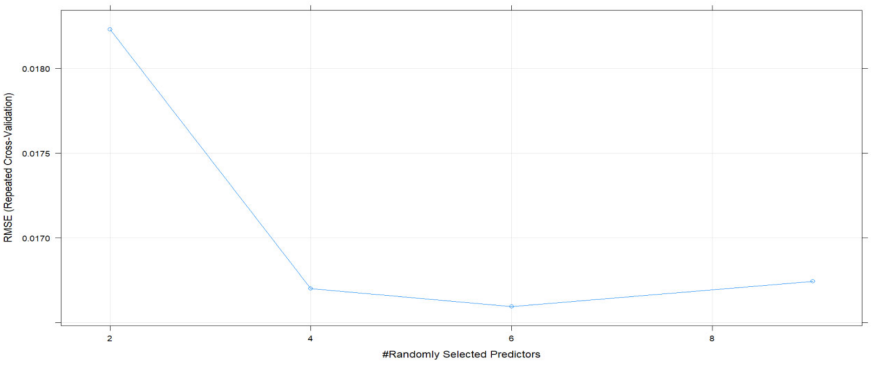
2.2.2 Random forest. RF is one of the most popular ML algorithms for solving classification and regression problems. RF easily adapts to the non-linear relationships, which makes it more likely to predict with greater accuracy than linear regression. RF is an ensemble of models using bagging, aimed at improving prediction accuracy and reducing variance through a collection of decision trees, especially when the predictions are negatively correlated or uncorrelated. In a RF, observations are randomly sampled with replacement (bootstrap) to create a bootstrap sample of the same size as the original dataset. Then, observations are repeatedly split using binary decision rules, characterized by a cut-off point on a specific predictor in the dataset. The predictor and its cut-off point are chosen to divide the observations into two groups (He *et al*., 2018). This process is repeated to create multiple decision trees, resulting in a "forest". The predictions from each tree are then combined to obtain a final, more accurate and robust prediction. In the case of classification, majority voting is used, while for regression, the one taken is the average.

Afterwards, the RF algorithm is applied to the Apple's American put options data to predict their price using the "train()" function from the "Caret" package in RStudio. For the construction of the algorithm, the "trainControl" function was first created to set the control parameters for the model training. In this function, repeated cross-validation is specified as the validation method, with 10 folds and three repetitions. Additionally, message printing during iteration ("verboseIter") is enabled, class probabilities ("classProbs") are disabled as it is a regression problem, and the summary function for the results ("defaultSummary") is defined.

Subsequently, the predictive regression model was trained using the RF method on the normalized training dataset. The previously created control function was used, and with "tuneLength", it was specified that four different models would be tested during the hyperparameter tuning process to achieve the best performance. The hyperparameters tested are 2, 4, 6 and 9.

In Figure 4, it can be observed how, depending on the structure adopted, the RF model will have a different RMSE value. The optimal hyperparameter to be used is 6, as it corresponds to the minimum RMSE value. This indicates that the model trained with this value has the best performance in terms of accuracy compared to the other hyperparameters tested.

Additionally, the variation in the error of the RF model as a function of the number of trees used can be seen in Figure 5. It shows that after 100 trees, the decrease in error is minimal. This suggests that adding more trees does not significantly improve the accuracy of predicting the price of Apple's American put options, thus helping to optimize the model's performance.



mtry	RMSE	Rsquared	MAE
2	0.01823029	0.9725568	0.007342439
4	0.01669969	0.9759134	0.006238818
6	0.01659429	0.9760293	0.006145728
9	0.01674400	0.9755849	0.006319601

Figure 4. RMSE graph of RF as a function of the number of hyperparameters (mtry or random of selected hyperparameters). Figure by authors

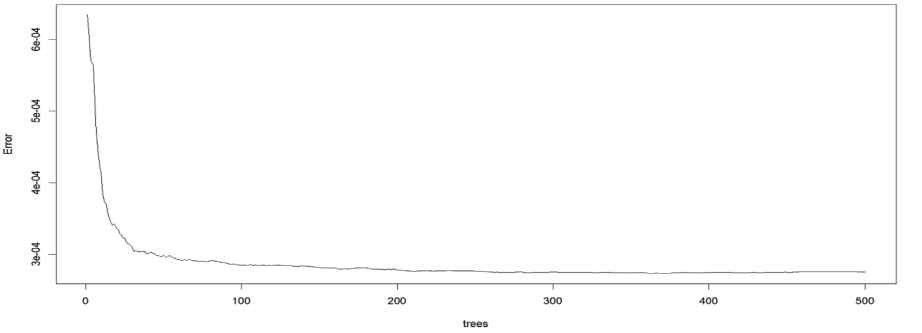


Figure 5. RF model error graph as a function of the number of trees. Figure by authors

Since the RF model cannot be visualized like decision trees, the importance of the variables has been analyzed using the "varImp" function (Table 3).

The analysis shows that the variables "STRIKE", "MAX_PAYOFF_PER_OPTION" and "PX_LAST_STOCK" are the three most influential in predicting the price of American put options. It is important to highlight the "STRIKE" variable, which is assigned a value of 100, due to its strong correlation with the target variable "PX_LAST_OPTION". This implies that small changes in this variable will significantly impact the RF model's predictions.

On the other hand, the three least important variables are "PX_VOLUME_STOCK", "DAYS_UNTIL_MATURITY" and "MAX_PAYOFF_PER_OPTION_ DAYS_UNTIL_ MATURITY". Their values are close to 0, with "PX_VOLUME_STOCK" even being assigned a 0. This suggests that the aforementioned variables will have minimal or insignificant impact on the prediction of the American put options price, and therefore, the RF model can exclude them without significantly affecting the results. Moreover, eliminating these variables will simplify the RF model's structure and reduce its complexity without compromising accuracy.

Table 3. Importance of the variables in the RF model

	Overall
STRIKE	100.000
MAX_PAYOFF_PER_OPTION	56.034
PX_LAST_STOCK	29.288
DIVIDEND_INDICATED_YIELD	17.651
USGOVT1YEAR_PX_LAST	4.136
VOLATILITY_1D	3.117
MAX_PAYOFF_PER_OPTION_DAYS_UNTIL_MATURITY	2.734
DAYS_UNTIL_MATURITY	1.808
PX_VOLUME_STOCK	0.000

Source(s): Table by authors

Since the data is normalized, it is returned to its original state to compare the RF model's price predictions with the original data of Apple's American put options.

The histogram (Figure 6) shows that most of the errors are concentrated around 0. The frequency of errors is higher for negative values, reaching its peak close to 0 and suggesting that the RF model tends to overestimate Apple's American put options price. However, even though errors greater than 50 in absolute terms are rare, it can be seen that there are more negative than positive values in that range. These large negative errors may occur when the actual price of the American put option is much lower than what the RF model predicts. This is possibly due to abrupt and unexpected movements in the financial market or sudden changes in volatility.

In order to test the accuracy of the RF model, its RMSE was calculated, yielding a value of 15.35954. This means that, on average, the RF model's predictions differ by approximately 15.36 units from the actual values.

Finally, when using stochastic volatility instead of historical one to evaluate prediction performance in a multidimensional context, the RMSE is 15.47832. This suggests that the model's performance is quite similar regardless of whether stochastic or historical volatility is used, indicating that the choice may have little influence on prediction accuracy.

2.2.3 MLP. Artificial neural networks are ML algorithms used for both classification and regression. MLPs are feedforward artificial neural networks with multiple fully connected layers that use non-linear activation functions for training.

A simple perceptron (LP) has limitations in its ability to map desired input-output relationships. This is because it only contains one neuron with adaptable synaptic weights and bias. The aforementioned limitation can be addressed by using an MLP neural network with more input data nodes and output layers interspersed with hidden layer nodes (Isabona *et al.*,

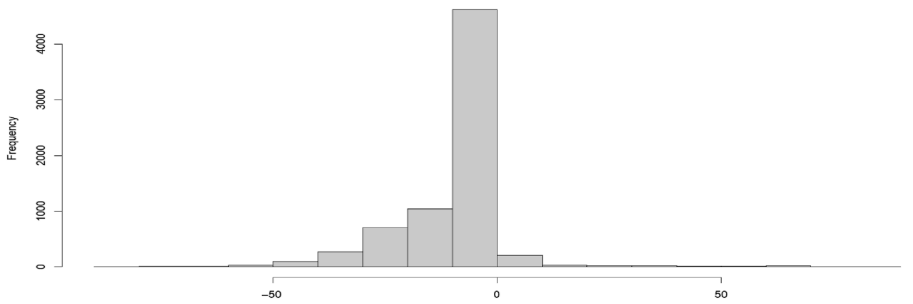


Figure 6. Histogram of RF prediction errors. Figure by authors

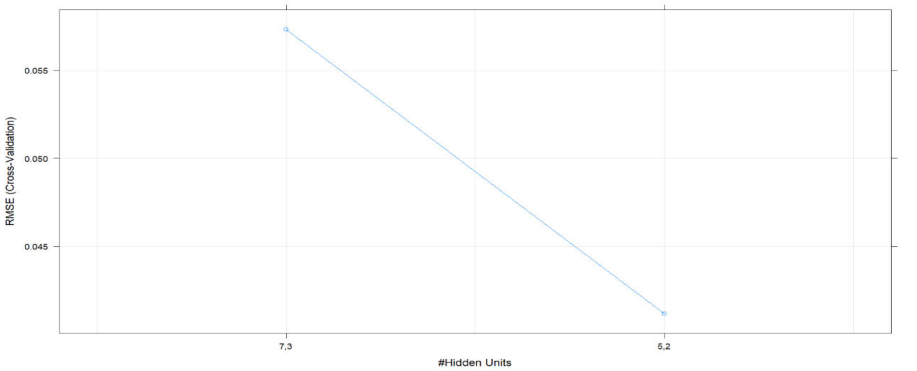
2022). In a MLP, each neuron in the hidden and output layers takes weighted inputs from the neurons in the previous layer. This allows for greater information processing and modeling capacity. Additionally, applying non-linear activation functions, such as the sigmoid function or Rectified Linear Unit (ReLU), enables the network to capture more complex and non-linear relationships.

Then, the MLP algorithm is applied to Apple's American put options data to predict their price. First, a grid set for the hyperparameter search is defined using "expand.grid". Specifically, considering the dimensions of the variables involved in the model, two possible sizes for the two hidden layers have been specified: "7, 3" and "5, 2". The MLP algorithm has been built using the "train()" function from the "Caret' package in RStudio. For training, the normalized data from the training set was used so that all features are on the same scale. "PX_LAST_OPTION" is the target variable, and all other variables are used to predict Apple's American put options price. Particularly, in the "method" argument, it is indicated that the algorithm is MLP. Additionally, three-fold cross-validation is applied to assess the model's performance, which involves splitting the data into three equal parts, training the model on two parts, and evaluating its performance on the remaining part. Moreover, data preprocessing is done by centering and scaling the features. For the hyperparameter search, the previously defined grid in "tuneGrid = mlp_grid" is used. Furthermore, "linOut = TRUE" is set to obtain a linear output instead of a logarithmic one. Finally, the MLP model's performance is evaluated using RMSE, and the ReLU activation function, a non-linear function, is employed, defined as $f(x) = max(0, x)$.

In Figure 7, it can be observed that, depending on the structure adopted, the MLP model will have a different RMSE value. The MLP model will have a structure with two layers, where the first layer contains five neurons and the second layer contains two neurons, as this minimizes the RMSE (Figure 8).

Since the data is normalized, it is returned to its original state to compare the MLP model's price predictions with the original data of Apple's American put options.

The histogram (Figure 9) shows that most of the errors are concentrated around 0. The highest frequency corresponds to negative values close to 0. This suggests that, overall, the MLP model tends to produce results close to the expected value and performs well. However, the fact that there are more negative errors indicates its tendency to overvalue Apple's American put options.



Size	RMSE	Rsquared	MAE
7,3	0.05732954	0.7257118	0.03090704
5,2	0.04117319	0.8606111	0.02457409

Figure 7. RMSE graph of MLP as a function of neuron structure (size or hidden units). Figure by authors

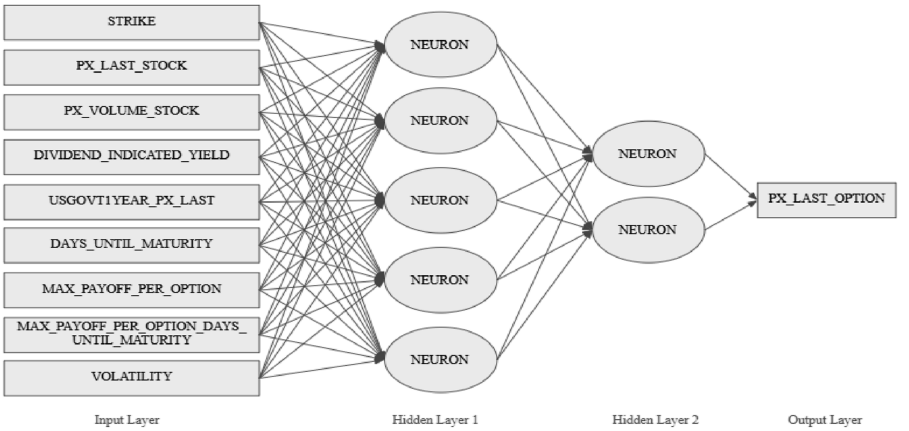


Figure 8. MLP structure. Figure by authors

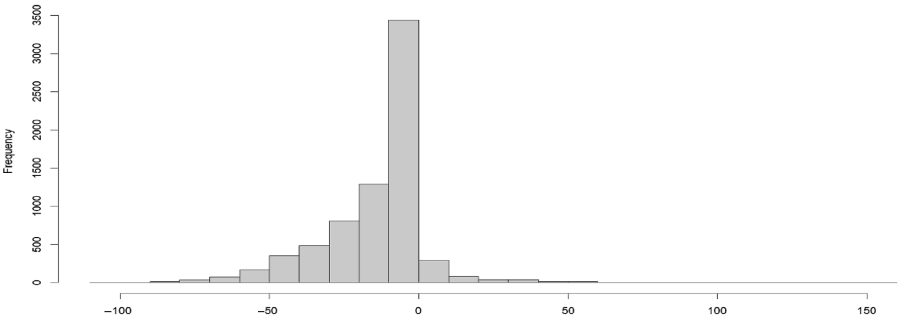


Figure 9. Histogram of MLP prediction errors. Figure by authors

In order to test the accuracy of the MLP model, its RMSE was calculated, yielding a value of 22.12773. This means that, on average, the MLP model's predictions differ by approximately 22.13 units from the actual values.

Finally, when using stochastic volatility instead of historical volatility to evaluate prediction performance in a multidimensional context, the RMSE is 22.7533. This implies that the model performs comparably whether stochastic or historical volatility is applied, suggesting that the selection between them may have a minimal effect on prediction accuracy.

2.2.4 CNN. CNNs are specialized deep NNs (deep learning) that analyze input data containing spatial structure. These deep NNs leverage the extraordinary computational power and large datasets available in many fields today more efficiently. CNNs are mainly used to solve various computer vision problems, using images as input data. In terms of their functionality, they first derive low-level representations, such as local edges and points, and then build higher-level representations, such as general shapes and contours. The name of these deep NNs comes from the application of convolutions in at least one of their layers, which is a type of linear mathematical operation. The use of convolutions replaces the general matrix multiplication employed by feedforward NNs (Montesinos López *et al*., 2022).

A typical CNN architecture (Figure 10) includes the following components. Convolutional layers apply filters to extract features from the input data, producing feature maps that capture essential patterns. This is followed by pooling layers, which reduce the spatial dimensions of

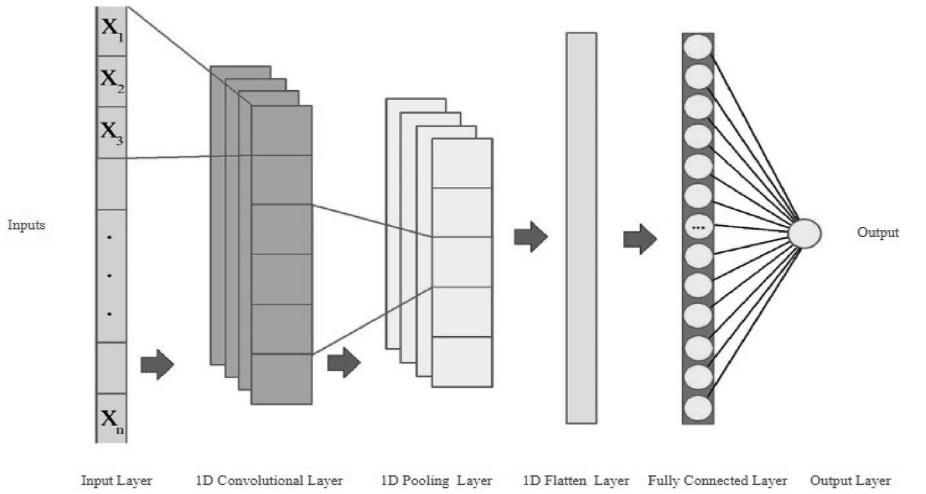


Figure 10. CNN structure. Figure by authors

the feature maps, retaining key information while lowering computational complexity and minimizing overfitting. The output of these layers is passed to a flattening layer, which converts the multidimensional data into a one-dimensional vector. This flattened vector is fed into fully connected layers, which learn complex combinations of features to make predictions. Finally, the output layer generates results, such as classification probabilities or continuous values for regression tasks, based on the processed data.

Next, the CNN algorithm is applied to Apple's American put options data to predict their price. Since CNNs expect three-dimensional inputs, an additional dimension has been added to the input data to fit the CNN model format. The CNN algorithm has been built using the "Keras" package in RStudio. The sequential model has been defined using "keras_model_sequential()", which adds sequential layers to the CNN model. The architecture of the CNN model begins with a one-dimensional convolutional layer that has 64 filters and a kernel size of two. The input for the layer is defined with the size of the inputs, and the ReLU activation function is used to learn complex patterns in the data. Later, a flattening layer converts data into two dimensions for the densely connected layers. The first dense layer has 32 units and ReLU activation, enabling the learning of more abstract data representations. The second layer has one unit and linear activation, making it the output layer that produces a continuous numerical prediction. To compile the model, MSE and the Adam optimizer were used to minimize prediction errors.

Table 4 summarizes the CNN model, providing an overview the NNs's structure and connections. It shows the architecture, including the layer type, output sizes, and the number of parameters, which totals 26,881.

To fine-tune the CNN model, the training data is used, and the "fit()" function is employed to train the CNN model for 100 epochs with a batch size of 16. During training, the CNN model adjusts the parameters so that the predictions are as close as possible to the actual observations of Apple's American put options, as recorded in the training data. In Figure 11, the CNN model's error at each epoch can be observed. Upon completing the training, the model is evaluated using the "evaluate()" function, yielding an error of 0.0001980907.

Since the data is normalized, it is returned to its original state to compare the CNN model's price predictions with the original data of Apple's American put options.

The histogram (Figure 12) shows that most of the errors are concentrated around 0. The highest frequency corresponds to negative values close to 0, with errors greater than −50 being

Table 4. Summary of the CNN model

Layer (type)	Output shape	Param #
conv1d_1 (Conv1D)	(None, 13, 64)	192
flatten_1 (Flatten)	(None, 832)	0
dense_3 (Dense)	(None, 32)	26,656
dense_2 (Dense)	(None 1)	33
Total params: 26,881 (105.00 KB)		
Trainable params: 26,881 (105.00 KB)		
Non – trainable params: 0 (0.00 Byte)		

Source(s): Table by authors

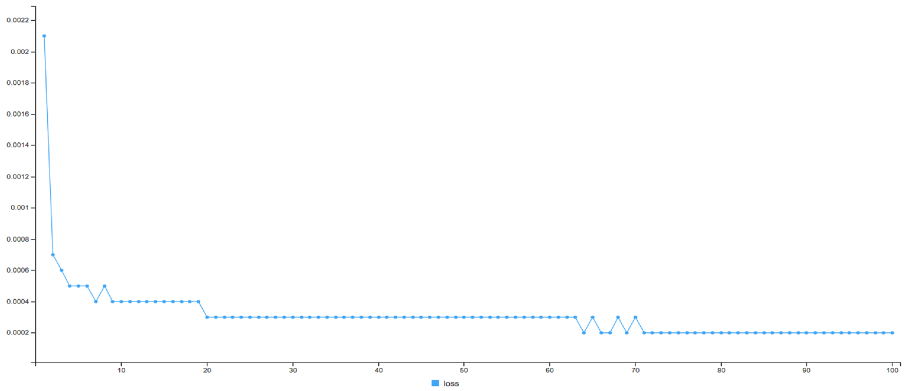


Figure 11. Graph of CNN model errors by Epochs. Figure by authors

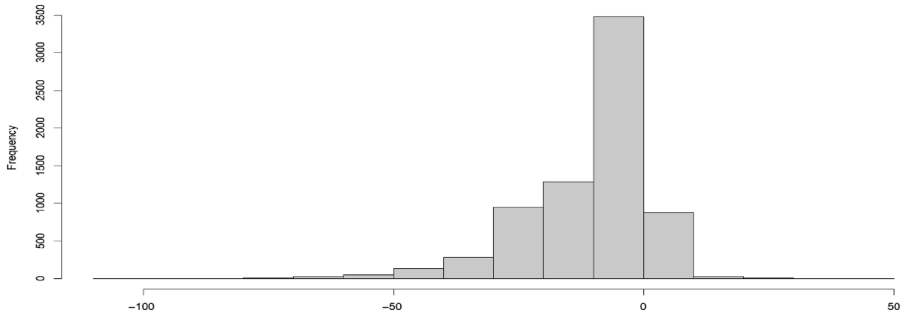


Figure 12. Histogram of CNN prediction errors. Figure by authors

minimal. Consequently, the CNN model tends to overestimate Apple's American put options price.

In order to test the accuracy of the CNN model, its RMSE was calculated, yielding a value of 16.44995. This means that, on average, the CNN model's predictions differ by approximately 16.45 units from the actual values.

Finally, when using stochastic volatility instead of historical volatility to evaluate prediction performance in a multidimensional context, the RMSE increased slightly to 17.78624 compared to 16.44995 with historical volatility. This indicates marginally better accuracy with historical volatility, but the overall similarity in performance suggests that the choice between the two has minimal impact on the model's predictive accuracy.

2.2.5 Longstaff-Schwartz. The L-S model (2001) is a key tool in valuing Aos, providing a numerical approach based on simulation with optimization. The model compares the potential benefit of exercising the American option at a given time (T) with the benefit that would be obtained by holding it without exercising.

The objective of the L-S model is to find the exercise rule that maximizes the value of the American option at each point in its lifetime. This process is carried out recursively, focusing on in-the-money paths to improve the model's efficiency and reduce computational complexity.

To determine the conditional expected value of holding the American option, L-S uses LSMC, which involves generating multiple paths representing the underlying assets' prices over time.

Then, a regression of future cash flows is performed based on relevant variables to estimate the conditional expected value of continuing with the American option. The resulting estimate from the regression provides an efficient measure of the conditional expectation, allowing the determination of the optimal exercise rule for the American option at each point in time.

Once the value of the American option is calculated for each path at time *T*, the values are discounted back to the initial moment ($T = 0$). This process involves adjusting the future cash flows by the appropriate discount factor to bring them to present value. Then, by averaging the values of each simulated path, the final price of the American option is obtained.

To apply the L-S model to the empirical analysis data, the "LSMonteCarlo" package from RStudio will be used (Beketov, 2013).

As previously explained, the variables to be used in this model are: "STRIKE", "USGOVT1YEAR_PX_LAST", "DIVIDEND_INDICATED_YIELD", "PX_LAST_STOCK", "VOLATILITY_1D", and "YEARS_UNTIL_MATURITY".

Since the "AmerPutLSM" function from the "LSMonteCarlo" package requires only a single value for each variable, different lists have been created. These lists contain the data from the Apple's American put options observations that make up the test set. Additionally, an empty list has been created to store the valuation results of Apple's American put options. This list will be used to compare the results with those obtained from the previously employed ML models.

To properly value the American put options with the available data, two modifications were made to the "AmerPutLSM" function.

First, it is adjusted to value the American put options on their expiration day. In this case, the American put option value will be the maximum between 0 and the difference between the strike price and the price of the underlying asset.

Second, a loop is created in the function to calculate the price of the American put options for each stock price of Apple contained in the test set observations. Price paths for the underlying asset are generated using the Geometric Brownian Motion (GBM) method. The results are then stored in the empty list created earlier.

Next, cash flows are calculated for each period in each path, taking into account the maximum between 0 and the difference between the strike price and the price of the underlying asset. A linear regression is then performed to estimate the value of the American put option, using the cash flows and the prices of the underlying asset in each period.

With the regression results, the control value is calculated to make exercise decisions in each period. This control value is compared with the intrinsic value of the American put option to determine whether it is optimal to exercise it at that specific moment. If the control value is greater than the intrinsic value, the decision is made not to exercise the American put option, and its future value is calculated. This process is repeated for all periods until expiration occurs.

Finally, the American put option price is calculated as the average of the discounted cash flows brought to the present time.

Once the prices of the American put options were obtained, they were compared with the real prices. In the histogram (Figure 13), it can be observed that the highest frequency of errors is in positive values close to 0. However, the larger errors are in negative values, reaching up to 300 units. Therefore, although the L-S model tends to make accurate predictions in most cases, it may struggle to predict accurately in certain extreme or unusual situations. This is due to the use of the GBM method, which does not account for heteroscedasticity and non-normal logarithmic returns. These limitations can lead to less accurate predictions in scenarios where volatility is high or the return distribution deviates significantly from a normal distribution, as is the case during extreme or unusual events in financial markets.

In order to assess the accuracy of the L-S model, its RMSE was calculated, yielding a value of 69.93252. This means that, on average, the L-S model's predictions differ by approximately 69.93 units from the actual values.

Finally, when using stochastic volatility instead of historical volatility to evaluate prediction performance in a multidimensional context, the RMSE increased to 74.32405 compared to 69.93252 with historical volatility. This difference indicates that the model achieves better accuracy with historical volatility, highlighting the importance of volatility choice.

3. Analysis of results and conclusions

In this analysis, the performance of the models is evaluated using the RMSE metric as an indicator of accuracy (Table 5).

When historical volatility is used, the KNN model has the best performance in terms of accuracy, with the lowest RMSE value of 11.28885. This result suggests that this approach is

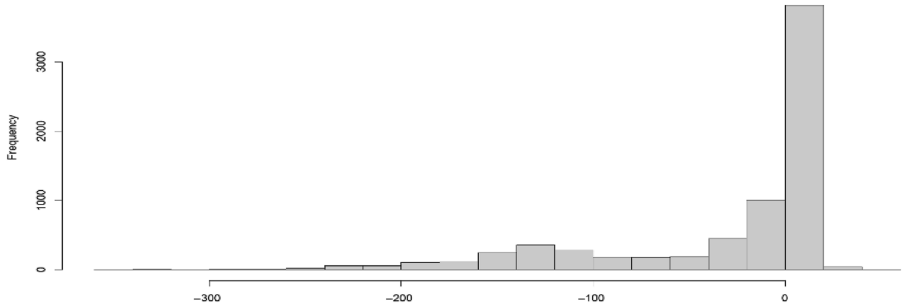


Figure 13. Histogram of Longstaff-Schwartz prediction errors. Figure by authors

Table 5. RMSE table of the models

Model	RMSE with historical volatility	RMSE with stochastic volatility
KNN	11.28885	11.40038
RF	15.35954	15.47832
MLP	22.12773	22.7533
CNN	16.44995	17.78624
L-S	69.93252	74.32405

Source(s): Table by authors

highly effective for predicting Apple's American put options price in this dataset. KNN stands out for its transparency, simplicity, interpretability, and computational efficiency.

On the other hand, the RF, MLP and CNN models have moderate performance in terms of accuracy when historical volatility is used, with an RMSE of 15.35954, 22.12773 and 16.44995, respectively. Although these algorithms can handle some data complexity, they do not appear to be as well-suited for this dataset as the KNN approach.

The L-S model presents a significantly higher RMSE than the ML algorithms when historical volatility is used, with a value of 69.93252. This suggests that the model does not fit well with the data for Apple's American put options compared to the ML approaches. This could be due to several reasons, such as the excessive simplification of the model's underlying assumptions and the limitations of the GBM method. As a result, the model struggles to capture the inherent complexity of financial market data.

It is also important to point out that even an RMSE of around 11.29, which may be considered low compared to the other models, is still significant in the context of valuing Aos. A difference of 11.29 in price prediction can have important financial implications, especially in a volatile market such as the Aos market. Even small discrepancies can translate into considerable valuation differences, affecting investment strategies and financial decisions.

When analyzing the models using stochastic volatility to evaluate their performance in a multidimensional context, the RMSE values are very similar to those obtained with historical volatility. This suggests both approaches capture the general behavior of the data. However, the most notable difference is in the L-S model, where the RMSE differs by approximately five.

In conclusion, highlight that the KNN, RF, MLP, and CNN algorithms, trained exclusively with historical financial market data, achieve a notable improvement in performance and accuracy in predicting the Aos price compared to the L-S model. The results show the potential of ML to tackle the more complex challenges faced by the financial industry today.

The L-S model, widely used in the valuation of Aos, shows limitations in capturing the complexity and dynamics of the financial market. This can lead to investment decisions that do not adequately reflect the financial market conditions. In contrast, ML algorithms, when trained with large amounts of historical data and using advanced techniques, can significantly improve the accuracy of predictions for the value of Aos.

Despite the remarkable advances achieved with these ML models, it is important to recognize that challenges and limitations still need to be addressed. Although the predictive values of the ML algorithms used in this work show a lower RMSE compared to the L-S model, implementing more sophisticated techniques is required to reduce it further. Accurate prediction is of great relevance in the context of Aos, where errors can have significant consequences, such as financial losses, arbitrage opportunities, impacts on investment strategies, and increased volatility in the financial market.

Moreover, it is worth noting that while the ML algorithms used have been shown to predict better than the L-S model, further research is needed to determine whether they achieve superior results compared to hybrid models, which have been applied by various authors in recent years.

Finally, this ML approach to price Aos faces the inability to derive hedging strategies and optimal exercise conditions. To address these gaps and enhance the ML models' practical relevance, future efforts could explore integrating deep reinforcement learning to identify optimal exercise policies or developing hybrid models that combine ML with traditional financial frameworks.

References

Anderson, D. and Ulrych, U. (2023), "Accelerated American option pricing with deep neural networks", *Quantitative Finance and Economics*, Vol. 7 No. 2, pp. 207-228, doi: 10.3934/QFE.2023011.

Arévalo De Pablos, A., Camacho Miñano, M. and Pérez-Hernández, F. (2024), "A hybrid model integrating artificial neural network with multiple GARCH-type models and EWMA for performing the optimal volatility forecasting of market risk factors", *Expert Systems with Applications*, Vol. 243, 122896, doi: 10.1016/j.eswa.2023.122896.

Becker, S., Cheridito, P. and Jentzen, A. (2020), "Pricing and hedging American-style options with deep learning", *Journal of Risk and Financial Management*, Vol. 13 No. 7, p. 158, doi: 10.3390/jrfm13070158.

Beketov, M.A. (2013), "LSMonteCarlo: American options pricing with Least Squares Monte Carlo method (Versión 1.0)", available at: https://cran.r-project.org/package=LSMonteCarlo

Black, F. and Scholes, M. (1973), "The pricing of options and corporate liabilities", *Journal of Political Economy*, Vol. 81 No. 3, pp. 637-654, doi: 10.1086/260062.

Cox, J.C., Ross, S.A. and Rubinstein, M. (1979), "Option pricing: a simplified approach", *Journal of Financial Economics*, Vol. 7 No. 3, pp. 229-263, doi: 10.1016/0304-405X(79)90015-1.

Dubrov, B. (2015), "Monte Carlo simulation with machine learning for pricing American options and convertible bonds", *SSRN*, doi: 10.2139/ssrn.2684523.

Feng, G., Liu, G. and Sun, L. (2013), "A nonparametric method for pricing and hedging American options", *Winter Simulations Conference (WSC)*, Washington, DC, United States, pp. 691-700, doi: 10.1109/WSC.2013.6721462.

He, L., Levine, R.A., Fan, J., Beemer, J. and Stronach, J. (2018), "Random forest as a predictive analytics alternative to regression in institutional research", *Practical Assessment, Research and Evaluation*, Vol. 23 No. 1, p. 1, doi: 10.7275/1wpr-m024.

Heston, S.L. (1993), "A closed-form solution for options with stochastic volatility with applications to bond and currency options", *The Review of Financial Studies*, Vol. 6 No. 2, pp. 327-343, doi: 10.1093/rfs/6.2.327.

Hoshisashi, K. and Yamada, Y. (2023), "Pricing multi-asset Bermudan commodity options with stochastic volatility using neural networks", *Journal of Risk and Financial Management*, Vol. 16 No. 3, p. 192, doi: 10.3390/jrfm16030192.

Isabona, J., Imoize, A.L., Ojo, S., Karunwi, O., Kim, Y., Lee, C.-C. and Li, C.-T. (2022), "Development of a multilayer perceptron neural network for optimal predictive modeling in urban microcellular radio environments", *Applied Sciences*, Vol. 12 No. 11, p. 5713, doi: 10.3390/app12115713.

Kanashiro Felizardo, L., Matsumoto, E. and Del Moral Hernández, E. (2022), "Solving the optimal stopping problem with reinforcement learning: an application in financial option exercise", *2022 International Joint Conference on Neural Networks (IJCNN)*, Padua, Italy, pp. 1-8, doi: 10.1109/IJCNN55064.2022.9892333.

Klidbary, S.H. and Arabameri, A. (2023), "A novel density-based KNN in pattern recognition", *13th International Conference on Computer and Knowledge Engineering (ICCKE)*, Mashhad, Iran, pp. 185-190, doi: 10.1109/ICCKE60553.2023.10326227.

Longstaff, F.A. and Schwartz, E.S. (2001), "Valuing American options by simulation: a simple least-squares approach", *The Review of Financial Studies*, Vol. 14 No. 1, pp. 113-147, doi: 10.1093/rfs/14.1.113.

Maidoumi, M., Zahid, M. and Daafi, B. (2023), "Pricing American option under exponential Levy Jump-diffusion model using Random Forest instead of least square regression", *Journal of Mathematical Modeling*, Vol. 11 No. 2, pp. 229-244, doi: 10.22124/jmm.2022.21756.1909.

Malpica, A. and Frias, P. (2019), "Valuation of an American option for the Spanish secondary reserve market using a machine learning model", *IEEE Trans. Power Sys.*, Vol. 34 No. 1, pp. 544-554, doi: 10.1109/TPWRS.2018.2859762.

Montesinos López, O.A., Montesinos López, A. and Crossa, J. (2022), "Convolutional neural networks", in Montesinos López, O.A., Montesinos López, A. and Crossa, J. (Eds),*Multivariate Statistical Machine Learning Methods for Genomic Prediction*, Springer, Cham, pp. 533-577, doi: 10.1007/978-3-030-89010-0_13.

White, A. and Hull, J. (1990), "Valuing derivative securities using the explicit finite difference method", *The Journal of Financial and Quantitative Analysis*, Vol. 25 No. 1, pp. 87-100, doi: 10.2307/2330889.

Corresponding author
María Coronado-Vaca can be contacted at: mcoronado@comillas.edu