International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

ANIN: Investment Analysis Environment

João Victor Magalhães Souza[1], Fernando de Souza Bastos[2]
[1,2]Institute of Exact and Technological Sciences, Federal University of Viçosa, Florestal, Minas Gerais, Brazil-35690000

Abstract— *Due to the availability and ease of handling our financial resources by applications, investments in variable income have become popular and have become, over the last few years, one of the first options for people looking for alternatives to monetize their capital. However, many applications that have tools of interest to investors are paid or depend on some level of knowledge in programming languages. In this context, we present the present work that provides a free interactive dashboard created from the Shiny package of the R language that applies the Machine Learning algorithm called XGBoost to predict the closing price of assets. Our application also provides asset price charts by periods of interest to the investor, price boxplot, candlestick charts and has a screen that provides results from the Markowitz Theory. Thus, we believe that the application created has the potential to help investors, especially beginners, to make more assertive decisions about investments in variable income on the São Paulo Stock Exchange. Finally, we present comparisons of the prediction results observed using the XGBoost algorithm and an artificial neural network used in the areas of artificial intelligence and deep learning called Long Short-Term Memory (LSTM). We observed that the results obtained by applying the XGBoost algorithm present better results in most cases.*

Keywords— *Investments, Machine Learning, Markowitz Theory, Stock Exchange, XGBoost.*

I. INTRODUCTION

When we want to make investments in variable income, it is important to analyze two aspects: the first refers to technical and quantitative analyzes on the behavior of the price of the asset that we want to invest and the second, no less important, is based on fundamental analysis, which is a study more in-depth information about the company and about the market related to the asset.

Both analyzes depend on a large amount of time and dedication on the part of the investor. There are, however, several software and applications that have methods capable of helping you. Still, most of these apps are paid or rely on technical knowledge to be used by interested people. n this way, this work proposes to create and make available, free of charge, an Investment Analysis Environment (ANIN) that includes a prediction tool for the closing price of the most traded shares on the Brazilian stock exchange B^3. Based on the work promoted by Hiransha (2018), whose purpose was to predict and compare the results obtained from the prediction of prices of assets traded on the Indian stock exchange via Deep Learning models, we implemented the price prediction from the application of the Machine Learning algorithm called XGBoost. Such an algorithm has a simpler implementation and was able to produce results like the Long Short-Term Memory (LSTM) model used by Hiransha (2018) in his article.

In addition, our application includes Time Series charts, boxplot and candlestick charts of prices of the most traded assets of B^3 and also allows the application of Markowitz Theory (MARKOWITZ, 1959). We believe, therefore, that our application can assist investors in making a more assertive investment decision in variable income.

II. MATERIAL AND METHODS

We use an R software package called BatchGetSymbols (PERLIN, 2020) which provides, from reading Yahoo Finance financial data, the daily opening, closing, highest price, lowest price, among other variables, of the assets since 2015. We then gathered data on the most traded assets on the Brazilian stock exchange since 2015, the list of which can be found in Appendix A. When specifying an asset to be monitored, the index of the responsible exchange and the period to be parsed, the library returns a dataset containing several attributes. The most important for the development of this project are:

- **ref.date:** variable responsible for holding the dates;
- **price.open:** variable responsible for showing the opening price of an asset on a given date;
- **price.high:** variable responsible for showing the highest price of an asset on a given date;
- **price.low:** variable responsible for showing the lowest price of an asset on a given date;
- **price.close:** variable responsible for showing the closing price of an asset on a given date;
- **price.adjusted:** variable responsible for showing the closing price of an asset after adjustments for splits and dividends on a given date;
- **volume:** variable responsible for showing the available trading volume of an asset on a given date.

We then perform some analysis to find missing data, typing errors and/or other problems that could make some important information contained in the data unfeasible. We made the necessary corrections and produced a database to be used in the construction of the ANIN.

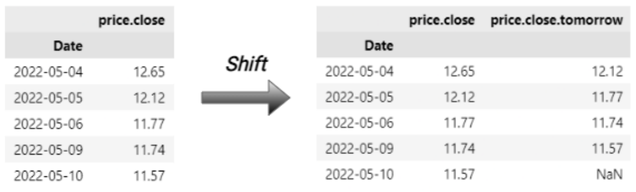
Even so, given the great complexity of our problem, which is the prediction of the closing price of an asset on the stock exchange, we need to complement our database by creating other variables to apply the Machine Learning algorithm. We then searched for some strategies in Feature Engineering area, for the creation of new variables to increase the efficiency of our predictions.

According to Brownlee (2020), Feature Engineering is understood as the process of transforming raw data into variables that best represent the underlying problem. We use this process to create features (or variables) to increase the quality of the information to be passed to the Machine Learning model. Basically, we have two aspects in this work for the process of creating new features:

- **Knowledge-Based Features:** to create these features, we evaluate existing attributes and external knowledge to create new variables. Below we present some variables created that will be of interest to us for the application of the XGBoost algorithm:
  - **Derivative:** Indicates the daily rate of change of an attribute;
  - **Moving Average and Standard Deviation:** The objectives of these features are to present the average and dispersive behavior, respectively, of the attributes considered at intervals of three days;
  - **Deltas:** Unlike previous approaches that are calculated singularly, that is, each new feature comes from just one existing input variable, this approach combines the differences between pairs of input variables. Some new input attributes created were:
    - **High-Low:** It presents the amplitude of variation in the price of an asset on a given day;
    - **High-Close:** Calculates the difference between the highest price and the closing price of an asset on a given date;
    - **Low-Close:** Calculates the difference between the lowest price and the closing price of an asset on a given date;
    - **Close-Open:** Calculates the ratio of the difference between the closing price and the opening price of an asset on a given day;
    - **High-Open:** Calculates the difference between the highest price and the opening price of an asset on a given date;
    - **Low-Open:** Calculates the difference between the lowest price and the opening price of an asset on a given date;
  - **Previous Closing Price:** The purpose of this feature is to map some trend in tomorrow's closing price based on previous days' closing price.
- **Purely Mathematical Variables:** This proposal looks exclusively at the data, its objective is to include information without a clear semantic proposal, that is, we can insert attributes created by the researcher to be evaluated by the algorithm during its execution. For each variable initially present in the database, we create other variables as mentioned below:
  - **Integral calculus:** Implements the calculation of the sum of a variable every 3 days;
  - **Polynomial Combinations:** We multiply pairs of variables and square each of the variables in the database.

Although quantity does not denote quality in Data Science, at the end of this process, we were able to include information that was not previously perceptible. We did not perform any redundancy analysis between the variables, since XGBoost, the algorithm we use for the prediction, already decides which variables are most relevant.

As we want to predict the closing price of an asset chosen by the user one day in advance, we create a new column in the database, called "price.close.tomorrow", which will be our target variable. The database has *N* rows each row *i* of this column received the closing price of the day *i+1*, $i \in \{1,2,3,\ldots,N-1\}$. In this way, it is easy to see that the Nth entry received no value. Our interest is fill in this entry, in such a way that the value found will be an estimate of the day's closing price *N+1*. The Figure 1 illustrates how this process, popularly known as variable shifting, is performed:



Date	price.close
2022-05-04	12.65
2022-05-05	12.12
2022-05-06	11.77
2022-05-09	11.74
2022-05-10	11.57

Date	price.close	price.close.tomorrow
2022-05-04	12.65	12.12
2022-05-05	12.12	11.77
2022-05-06	11.77	11.74
2022-05-09	11.74	11.57
2022-05-10	11.57	NaN

Fig. 1. Target creation process.

In this way, we have a dataset ready to be used. XGBoost is a Machine Learning algorithm based on decision tree models that uses the concepts of ensemble and gradient boosting to build an optimized algorithm (CHEN; GUESTRIN, 2016). Unlike the conventional Decision Trees approach, this algorithm produces several Decision Trees (ensembles) and has techniques to minimize the error associated with the predictions made in each of the generated trees (gradient boosting).

As, in general, different Decision Trees are generated, XGBoost offers a much more significant generalization capacity when compared to simpler tree strategies, precisely because it operates on distinct subsets of data, allowing the reconsideration of the importance of a feature during the tree creation process.

Furthermore, with the use of XGBoost, additional steps in the data pre-processing process such as input and/or substitution of values and data normalization do not need to be performed, unlike when using Neural Network algorithms, such as Long Short -Term Memory (LSTM). In this case, it is necessary to carry out these pre-processing steps and adapt the data format to the Time Series format. With XGBoost we can create several models for each user interaction with the system, a fact that is not feasible with an LSTM due to the delay in training time.

Finally, we emphasize the importance of our work, since, traditionally, most problems that involve prediction of values in stock exchanges or commercial applications make use of Neural Network algorithms, especially the Recurrent ones, and our article presents a simpler algorithm of Machine Learning, with very similar or better results than those produced with Neural Network algorithms, according to comparisons presented in the results section.

To measure the quality of predictions made by the XGBoost model compared to the LSTM model, we used the Mean Absolute Error (MAE), the Mean Squared Error (MSE) and the Mean Absolute Percentage Error (MAPE) of the prediction. Given an input of size *n*, the Mean Absolute Error (MAE) is the absolute mean of the difference between the predicted value and the actual value and can be written, mathematically, as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |(Pi - Ri)| \qquad (1)$$

where *Pi* is the predicted value and *Ri* is the actual value. The Mean Squared Error (MSE) is a more sensitive error metric to predictions that are further from the true value:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Pi - Ri)^2 \qquad (2)$$

In addition, we have the Mean Absolute Percentage Error (MAPE) of the prediction, calculated as:

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{(Pi - Ri)}{Ri}\right| \qquad (3)$$

The Investment Analysis Environment (ANIN) that we created was built in the R language using the shiny package, created by Chang et al. (2021). To apply the XGBoost algorithm, whose library is implemented in Python (CHEN; GUESTRIN, 2016), t was necessary to use the R package called reticulate created by Ushey, Allaire and Tang (2022). uch a package communicates the R language with Python code. Our environment also includes pages with graphs that show variations in the prices of the most traded assets in B^3 and a page with the Markowitz Theory implemented for user evaluation.

Finally, it is important to remember that the exact forecast of the closing price, for the investor, may not be as valuable as the trend information of this price, that is, knowing if this price will appreciate, depreciate or remain stable the next day. That said, we also propose a calculation of the percentage of accuracy of the trend of the predictions made by XGBoost. We consider *i* to be any business day, *Ri* to be the actual closing price of a given asset on day *i* and *Pi* to be the prediction made for day *i*, we can, then calculate the assertiveness of the trend as follows:

```
hits = 0
mistakes = 0
if ($R_{i-1} – R_i \leq 0$) and ($R_{i-1}$ - $P_i \leq 0$) then
    hits = hits + 1      #Uptrend or stability hit.
else
    if ($R_{i-1} – R_i > 0$) and ($R_{i-1}$ - $P_i > 0$) then
        hits = hits + 1      #Downtrend hit.
    else
        mistakes = mistakes + 1    #Wrong trend.
```

III. Results

In the link https://dashboard-st5shwubpq-uc.a.run.app/, we can view the Investment Analysis Environment (ANIN). on the page, choose a certain asset and obtain the prediction of the asset's closing price for the next business day as a result. In addition to being able to verify the percentage of hits of the trend of the last ten days and the margin of error of the prediction.

Before releasing our application, we performed some analysis to evaluate the results. We then divide the data set into training and testing, as percentages,

- **Training set:** 90% of the database;
- **Test set:** 10% of the database.

To apply the XGBoost algorithm with the xgboost library we use the following hyperparameters,

- **n-estimators:** 1500;
- **learning-rate:** 0.05;
- **max-depth:** 12;
- **subsample:** 0.8.

We present below the prediction results of the assets B3SA3.SA, VALE3.SA and BBAS3.SA obtained after the application of XGBoost and the LSTM algorithm, the latter is also implemented in Python and was used via the TensorFlow library of Abadi et al. (2015).

In the charts of Figures 2, 3 and 4 we visualize the real price and the predictions obtained via XGBoost and LSTM from August/2021 to April/2022. We also consider, in Tables 1 to 3, *P* as the set of daily predicted values and *R* as the set of real values of the asset, both in the same period shown in the graphs.

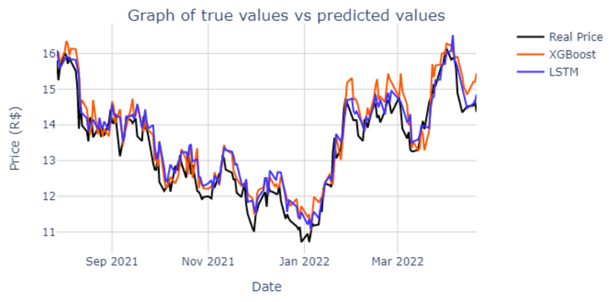


Fig. 2. Comparison between the real value and the predictions of the closing prices of the asset B3SA3.SA using XGBoost and LSTM algorithms.

We can see in Figure 2 that the price values estimated by XGBoost and LSTM are visually close to the real value and, in general, follow the trend of the price change movement. Furthermore, in Table 1 we observe small values of MAE, MSE and MAPE for the two algorithms.

TABLE I. Statistics for comparison of predictions made for asset B3SA3.SA via XGBoost and LSTM.

Predictions metrics for B3SA3.SA.				
Algorithm	**MAE**	**MSE**	**MAPE**	**R^2**
XGBoost	0.53	0.39	3.12	0.78
LSTM	0.34	0.18	2.53	0.89

On the other hand, in Figure 3, it is possible to notice that XGBoost does not follow the period between September/2021 and beginning of March/2022 and that LSTM presented a more assertive correspondence in relation to the behavior of the curve of the real price.

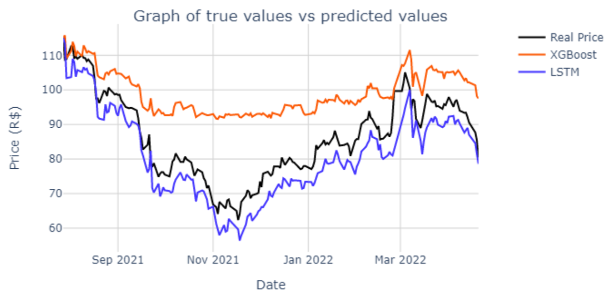


Fig. 3. Comparison between the real value and the predictions of the closing prices of the asset VALE3.SA using XGBoost and LSTM algorithms.

In addition, observing the MAE, MSE, MAPE and R^2 metrics from Table 2, the discrepant difference between the values obtained using LSTM and XGBoost is visible. That is, in the case of the asset VALE3.SA, the LSTM algorithm presented better results than XGBoost.

TABLE II. Statistics for comparison of predictions made for asset VALE3.SA via XGBoost and LSTM.

Predictions metrics for VALE3.SA.				
Algorithm	**MAE**	**MSE**	**MAPE**	**R^2**
XGBoost	11.76	183.54	13.07	-0.27
LSTM	5.12	36.64	7.10	0.74

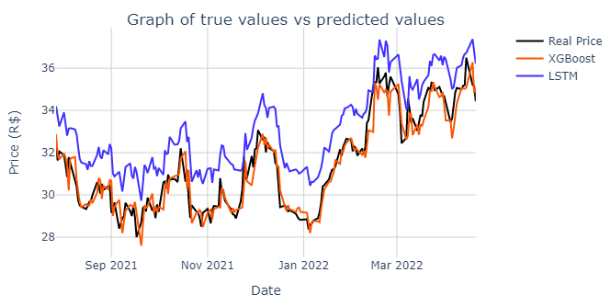


Fig. 4. Comparison between the real value and the predictions of the closing prices of the asset BBAS3.SA using XGBoost and LSTM algorithms.

For the asset BBAS3.SA, we can visually observe better results from XGBoost, when compared to LSTM, as shown in Figure 4. Note that the XGBoost prediction curve more accurately follows the actual price curve when compared to the LSTM prediction curve. Furthermore, Table 3 corroborates the fact that XGBoost's predictions better correspond to the reality of the real behavior of the asset with a low MAE, MSE and high value of R^2 which is not occurs when we analyze the values obtained using the LSTM.

TABLE III. Statistics for comparison of predictions made for asset BBAS3.SA via XGBoost and LSTM.

Predictions metrics for BBAS3.SA.				
Algorithm	**MAE**	**MSE**	**MAPE**	**R^2**
XGBoost	0.50	0.42	1.53	0.91
LSTM	1.69	3.21	5.16	0.34

We can notice that some results favor the use of LSTM and others of XGBoost. Therefore, the following data seeks to define which approach generally excels in the MAE, MSE, MAPE and R^2 metrics, considering all 67 assets listed in Appendix A.

TABLE IV. Comparison of the XGBoost and LSTM algorithms applied to the 67 most traded assets on the Brazilian Stock Exchange.

Predictions metrics for all assets.				
Algorithm	**Minor MAE**	**Minor MSE**	**Minor MAPE**	**Major R^2**
XGBoost	40 assets	41 assets	41 assets	41 assets
LSTM	27 assets	26 assets	26 assets	26 assets

Looking at the Table 4, it is possible to notice a much superior performance of XGBoost in all evaluated metrics. These analyzes indicate that, for most of the assets used in this study, the implementation of an XGBoost model is more efficient than the use of an LSTM model. However, whenever possible, we recommend a hybrid approach, given the particularities of each asset and each Machine Learning model. In addition, Deep Learning models tend to present better and more consistent results with a greater volume of data. XGBoost shows good results, even in small samples.

Finally, we performed the prediction of the closing value of all assets for a period of 10 business days (11/06/2022 to 24/06/2022) and calculated the percentage of trend hit by XGBoost. Below, we present the results of the 10 assets with the highest percentage of hits in the period.

TABLE V. Top 10 stocks with the best trend hit percentages by XGBoost in the analyzed period.

Trend hit for all assets.		
Ranking	**Asset Name**	**Trend Hit**
1º	HYPE3.SA	80.00%
2º	CSAN3.SA	80.00%
3º	B3SA3.SA	80.00%
4º	PCAR3.SA	70.00%
5º	MULT3.SA	70.00%
6º	MGLU3.SA	70.00%
7º	LREN3.SA	70.00%
8º	ENBR3.SA	70.00%
9º	VIVT3.SA	70.00%
10º	TOTS3.SA	70.00%

The Table 5 then shows that, for example, 80.00% of the predictions for the asset HYPE3.SA that pointed to an increase, stability or decrease for the following day were assertive. Another interesting point is that, for about 52% of assets, our XGBoost model hit at least 50% trend in the analyzed period.

IV. Source-Code

All the source code for the construction of the ANIN as well as the files for generating the analysis can be found at: https://github.com/JoaoVictorMagalhaesSouza/ANIN.

Appendix

All assets used		
ABEV3.SA	EGIE3.SA	MRVE3.SA
B3SA3.SA	ELET3.SA	MULT3.SA
BBAS3.SA	ELET6.SA	PCAR3.SA
BBDC3.SA	EMBR3.SA	PETR3.SA
BBDC4.SA	ENBR3.SA	PETR4.SA
BBSE3.SA	ENEV3.SA	PRIO3.SA
BEEF3.SA	ENGI11.SA	QUAL3.SA
BRAP4.SA	EQTL3.SA	RADL3.SA
BRFS3.SA	EZTC3.SA	RAIL3.SA
BRKM5.SA	FLRY3.SA	RENT3.SA
BRML3.SA	GGBR4.SA	SANB11.SA
CCRO3.SA	GOAU4.SA	SBSP3.SA
CIEL3.SA	GOLL4.SA	SULA11.SA
CMIG4.SA	HYPE3.SA	SUZB3.SA
COGN3.SA	ITSA4.SA	TAEE11.SA
CPFE3.SA	ITUB4.SA	TIMS3.SA
CPLE6.SA	JBSS3.SA	TOTS3.SA
CSAN3.SA	JHSF3.SA	UGPA3.SA
CSNA3.SA	KLBN11.SA	USIM5.SA
CVCB3.SA	LCAM3.SA	VALE3.SA
CYRE3.SA	LREN3.SA	VIVT3.SA
ECOR3.SA	MGLU3.SA	WEGE3.SA
	MRFG3.SA	

ACKNOWLEDGMENT

We thank Google LLC for the financial support of this project.

REFERENCES

[1] ABADI, Martín et al. TensorFlow: Large Scale Machine Learning on Heterogeneous Systems. [S.l.: s.n.], 2015. https://www.tensorflow.org/https://www.tensorflow.org/. Software available from tensorflow.org.

[2] BROWNLEE, Jason. Discover Feature Engineering, How to Engineer Features and How to Get Good at It. Machine Learning Magistery, 2020.

[3] CHANG, Winston et al. shiny: Web Application Framework for R. [S.l.: s.n.], 2021. https://CRAN.R-project.org/package=shinyhttps://CRAN.R-project.org/package=shiny. R package version 1.7.1.

[4] CHEN; GUESTRIN. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. In: p. 785.

[5] HIRANSHA. NSE Stock Market Prediction Using Deep-Learning Models. International Conference on Computational Intelligence and Data Science, 2018.

[6] MARKOWITZ, Harry M. Portfolio Selection: Efficient Diversification of Investments. [S.l.: s.n.], 1959. https://cowles.yale.edu/sites/default/files/files/pub/mon/m16-all.pdf. Monography (Economics), Yale University, New York, United States of America.

[7] PERLIN, Marcelo. BatchGetSymbols. Brasil: [s.n.], 2020. https://cran.r-project.org/web/packages/BatchGetSymbols/BatchGetSymbols.pdfhttps://cran.r-project.org/web/packages/BatchGetSymbols/BatchGetSymbols.pdf.

[8] USHEY, Kevin; ALLAIRE, JJ; TANG, Yuan. reticulate: Interface to ’Python’. [S.l.: s.n.], 2022. https://CRAN.R-project.org/package=reticulatehttps://CRAN.R-project.org/package=reticulate. R package version 1.25.