Doubly Predicted Time Series

A brief explanation about the method and its application

Mário Henrique Figlioli Donato

January 12, 2024

(Disclaimer: This report was intentionally written with LIMITED information about the method. It is solely intended to be part of my résumé and it is not a guide on how to implement such method.)

1 What DPTS is

The Doubly Predicted Time Series (DPTS) is a method to predict the value of random time series (e.g. the price of stocks). It was inspired by the following question:

> Given the data of a set of different (but correlated) random time series in a certain period of time (e.g. the prices of stocks in a portfolio for a month), is it possible to predict the upper and lower bound values of a target time series in the next period of time (e.g. the pessimistic and optimistic price predictions of a certain stock for next month)?

The answer is yes, within a certain accuracy (about 80% in current testings with PETR4), and a typical DPTS output is in Figure 1.

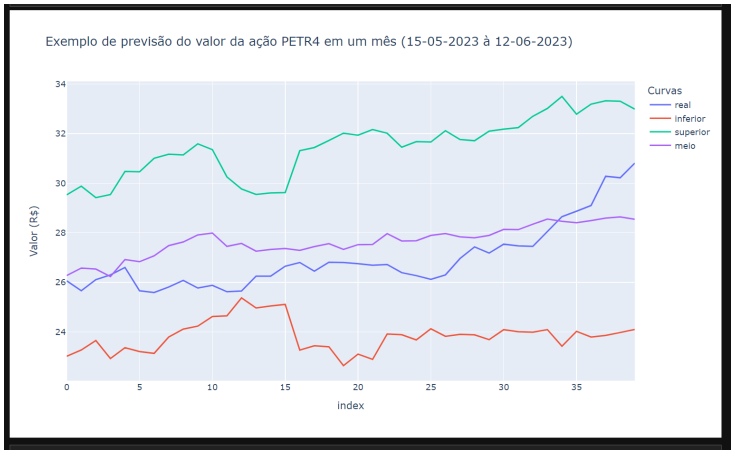

Figure 1: A typical output of DPTS method. The red and green lines are (respectively) the pessimistic and optimistic predictions of the PETR4 prices. The purple line is the arithmetic average between the pessimist and optimistic predictions, and the blue line is the real price of PETR4. (Data from 05-15-2023 to 06-12-2023).

2 The construction of the method

The model used to define DPTS is quite simple. It is defined by a set of different linear functions, chosen according to the current volatility class (i.e., a scale that measures how rapid are the variations of a time series) of

the target time series, as

$$y_{n+1} = A(v[x_n])X_n + b(v[x_n]), \tag{1}$$

where x_n is a vector (list) filled with the values of the target time series in a given period of time (e.g. prices of PETR4 during a month), X_n is filled with the values of the different time series in the same period of time (e.g. the stock prices in the portfolio during a month), and $v[x_n]$ is the volatility of x_n. There is more than one way to define "volatility" for a time series. In the presented method, it is defined with its *Fourier Transform*, quantifying how relevant the higher frequency spectrum is:

$$v[x_n] := 100\% \times \frac{\sum_{i \in 25\% \text{ higher frequencies}} |\text{fft}[x_n]_i|}{\sum_{i \in \text{all frequencies}} |\text{fft}[x_n]_i|}. \tag{2}$$

Then, it is possible to define volatility classes by setting intervals of volatility.

Also, $A(v[x_n])$ and $b(v[x_n])$ are the matrices that generate the prediction y_{n+1}, which has got information about the lower and upper bound predictions of x_{n+1}. Finally, the *stochastic gradient descent* method is used to fit the model.

How to extract the information in y_n about the upper and lower bound predictions and the used *loss-function* are intentionally omitted in this report.

3 Using DPTS to predict the value of PETR4

To test the method in a practical application, a study on the price prediction of PETR4 was done. Using the open source Python library "yfinance[1]" (which uses the "Yahoo Finance" API), we collected around 2 years of opening and closing market data of HAPV3, BBDC4, ITSA4, PETR4, B3SA3, RAIZ4, ABEV3, LREN3, CMIG4. We treated the data using the "Pandas" library, shaping it according to the vector X_n (with 20 days of data). Also, we calculated all possible volatility values of PETR4 beforehand and we defined 3 volatility classes:

$$\begin{aligned} 0 &: v[x_n] \leq 4.4\%, \\ 1 &: 4.4\% < v[x_n] \leq 6.2\%, \\ 2 &: 6.2\% < v[x_n]. \end{aligned} \tag{3}$$

These values were chosen so that there is approximately the same number of elements from the original dataset in each class.

[1] https://pypi.org/project/yfinance/

The stochastic gradient descent method was used to fit the model (using a suitable random train-test split strategy) and there are 4 examples below (Figures 2-5) of DPTS outputs: a successful one, two reasonable ones, and a bad one.

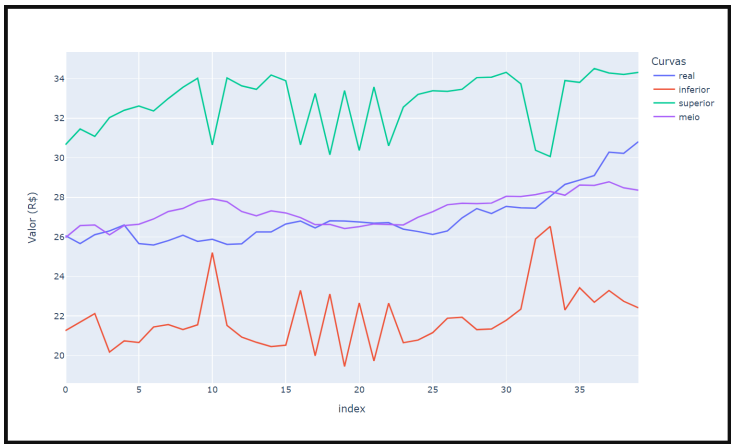

Figure 2: Example of a successful DPTS output.

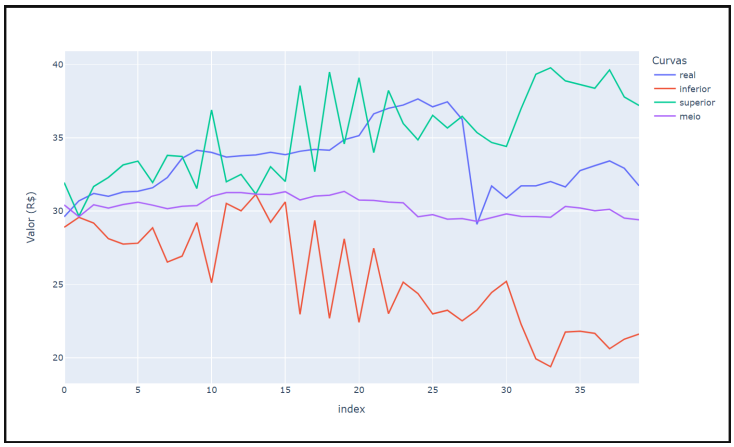

Figure 3: Example of a reasonable DPTS output.

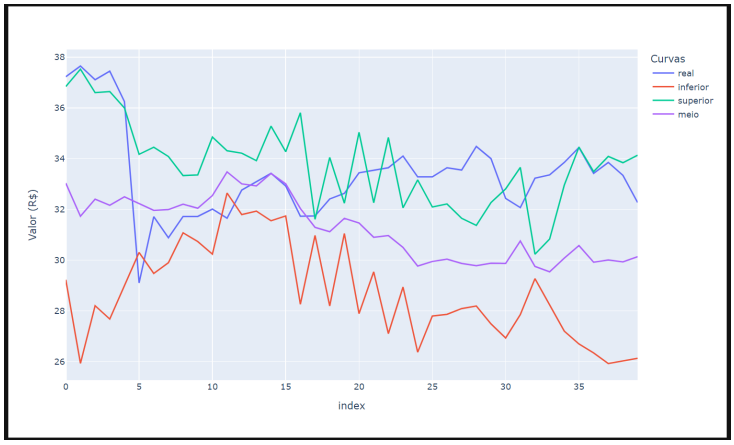

Figure 4: Other example of a reasonable DPTS output.

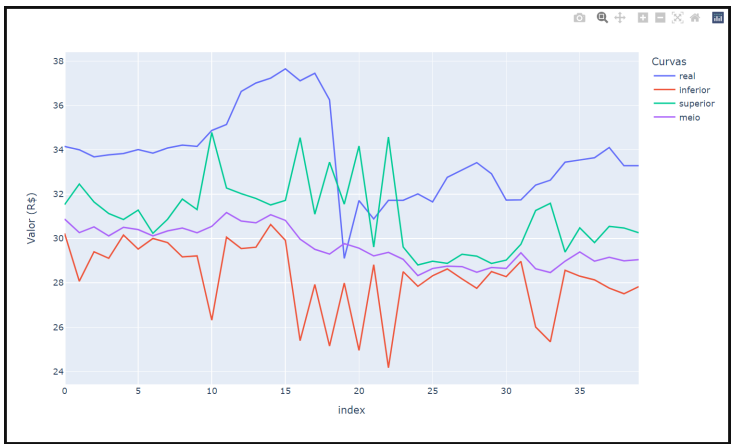

Figure 5: Example of a bad DPTS output.

In a successful output, all real prices of PETR4 are inside the predicted upper and lower bounds. In a reasonable one, most of the real data are inside the predicted bounds and/or the predicted curves (i.e., the optimist, pessimistic and average curves) are capable of reasonably describing them. In a bad one, the prediction does not agree with the real data. The labels in these graphs are the same as in Figure 1.

4 Results and its current accuracy

The accuracy of DPTS is defined by counting (in a relative 0 to 1 scale) how many real data points are within the predicted upper and lower bounds:

$$\text{accuracy} = \frac{\#\text{ of points within the bounds}}{\#\text{ of points}} \tag{4}$$

Below, there is the graph of all possible accuracy values found (also including the overlap of data in the test sample, hence the large number o points).

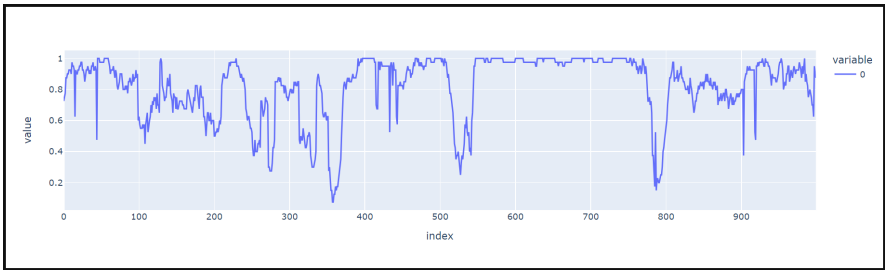

Figure 6: General accuracy score.

The average accuracy found was 0.82 (or 82%) with standard deviation of 0.20 (or 20%).

According to current analysis, it is likely that the "dips" in the accuracy graph happen when the volatility are near the bounds of a volatility class or when the volatility is very high or low (which is beyond the common dynamics of the stock market). Below, there is the graph showing the evolution of the volatility in time. Note that dips above tend to align when there is a rapid change in volatility.

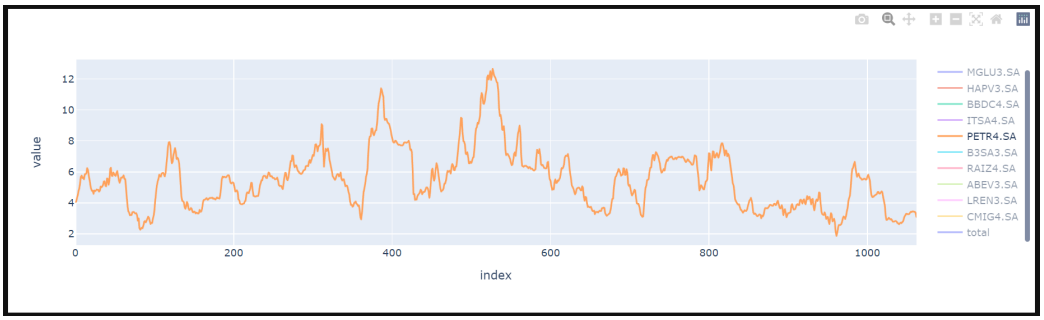

Figure 7: Evolution of the Volatility of PETR4 in time

5 Final considerations

DPTS is still a work-in-progress method to predict the behavior of Random Time Series. However, it is already capable to predict the value of PETR4 with 80% average accuracy.

Details on how to implement the DPTS method were intentionally omitted. Its implementation and use are currently reserved only to the Author of this report.