Forecasting time series by using deep learning techniques

[1] Oscar Vega
Yachay Tech
HPC Course
oscar.vega@yachaytech.edu.ec
Overleaf link https://es.overleaf.com/8318441177qjccnkrfwhrv

***Abstract*—This document presents the importance of forecasting and time series in the banana exports industry. Propose the sliding window method and the series_to_supervised function to convert a data vector into a data matrix in order be used as inputs and outputs for a neural network. Finally, presents the metric to measure time series accuracy, the two neural networks models: LTM squential and LSTM parallel, and HPC considerations.**

***Index Terms*—Deep Neural Networks, Time series, Sliding window, Forecasting, LSTM.**

I. Introduction

Forecasting and time series analysis are relevant topics because they can help to explore and analyze different characteristics of data related to some fields such as traffic, finance, engineering, complex networks, and so on [1]. The aim of time series analysis is to study the path observations of time series and build a model to describe the structure of data and then predict the future values of time series [2]. For example, predicting project costs can help individuals and organizations reduce costs and schedule projects.

Deep neural networks have received an increasing amount of attention in time series analysis. Studies that use deep neural networks for time-series data show great results and provides an extensive vision in increasing the usage of neural networks architectures in time series data [3]. Due to the capability to extract higher order features, deep neural networks can identify complex patterns within and across time series, and can do so from data sets of raw time series with considerably less human effort [4].

II. Problem

Deep neural networks techniques have caught the attention of economic field. On the last years, some techniques like Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM) model, between others, have been deployed for predicting time series data, and have caught the attention of many economic agents which are interesting in export forecast because it provides valuable information and can support economic policy makers, monetary authorities, and exporting firms in making decisions [5]. Therefore, in this projects it will develop two deep neural networks, LTM and MLP in order to know which offers best predictions with respect to lower forecast errors and higher accuracy of forecasts by applying a specific data set of the number of monthly exports made by Ecuador between the years 2012-2020.

Identify applicable funding agency here. If none, delete this.

This dataset was chosen because for 25 years Ecuador has been the leading exporter of banana in the world. The banana trade represents for the country, after oil, the second source of income for its economy and, consequently, contributes significantly to the long process of its development [6]. However, in recent years, banana exports have undergone a variation, becoming the second product of non-oil exports [7]. According to [8] the exportation of this fruit until October 2020 were of 260.6 million of boxes, which represents 8.5% more than the exportation made in the same period of 2019 and higher figure in comparison with the figures reached 2016, 2017 and 2018.

III. Univariate Forecasting

A time series is a set of observations measured sequentially through time. The term ”univariate” refers to a time series that consists of single observations recorded sequentially through time (every hour, day, week, month, year, or any other regular interval) without taking into account the effect of the other variables.

This project pretends to use deep neural networks and data from an univariate time series which are stored in a one-dimensional vector of 1x248 observations or samples. Then, the time series data will be reshaped as a matrix for a supervised learning problem, because it is necessary instruct to the neural network what output should be obtained from each specific input value.

Therefore, the matrix of n-inputs and n-outputs, through which the neural network will be trained, tested and evaluated, will use the sliding window method because it can be used to make predictions and to split and reshape the vector data into a matrix.

A. Sliding window algorithm

Sliding Window is a temporary approximation over the actual value of the time series data. The size of the window and segment increases until we reached the less error approximation. After selecting the first segment, the next segment is selected from the end of the first segment. The process is repeated until all time series data are segmented. For this research, the sliding window size is equal to 12, as on Fig1, and it accumulate the historical time series data used to predict next year of banana exports.

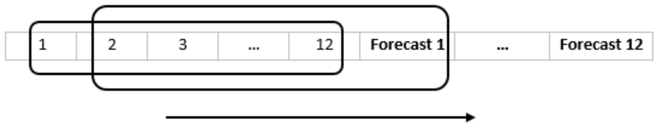


Fig. 1. Window size = 12.

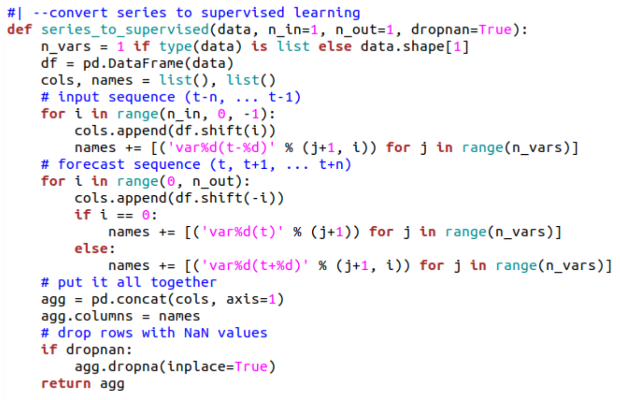
Each number (1, 2, 3 ... 12) represents monthly observation of time series data of month 1, 2, 3. . . .12 respectively. Initially window has covered from 1 to 12 which represents that 12 months historical data are being used to predict the next month of exports, then window slides right side by one month to cover another 12 month observations, including the recent prediction made, and the process will end when we have done 12 predictions which will represent the next year exports.

IV. Implemented Algorithms

A. Sliding window algorithm to convert time series data into a supervised problem

All the project will be implemented on a Linux Operative System and Python3. For develop this first part that correspond to the data preprocessing it is used pandas library which is used for data analisys.

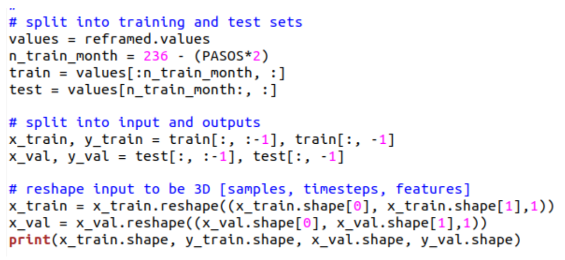
The supervised_to_learning function essentially shifts the vector data n steps in time and take three parameters; the first one is the time series data, the second one is the number of inputs or steps that is equal to the size window, and the third one will be the number of targets for the neural network. .



```
#| --convert series to supervised learning
def series_to_supervised(data, n_in=1, n_out=1, dropnan=True):
    n_vars = 1 if type(data) is list else data.shape[1]
    df = pd.DataFrame(data)
    cols, names = list(), list()
    # input sequence (t-n, ... t-1)
    for i in range(n_in, 0, -1):
        cols.append(df.shift(i))
        names += [('var%d(t-%d)' % (j+1, i)) for j in range(n_vars)]
    # forecast sequence (t, t+1, ... t+n)
    for i in range(0, n_out):
        cols.append(df.shift(-i))
        if i == 0:
            names += [('var%d(t)' % (j+1)) for j in range(n_vars)]
        else:
            names += [('var%d(t+%d)' % (j+1, i)) for j in range(n_vars)]
    # put it all together
    agg = pd.concat(cols, axis=1)
    agg.columns = names
    # drop rows with NaN values
    if dropnan:
        agg.dropna(inplace=True)
    return agg
```

Fig. 2. Sliding window algorithm to convert time series data into a supervised problem

The lasts steps to get ready the data will be; divide the reframed data set into 212 and 36 sets for training and testing respectively, and finally reshape all this data as a tensor of (samples, time steps, features) as on Fig.3 in order to pass it through the neural network.



```
..
# split into training and test sets
values = reframed.values
n_train_month = 236 - (PASOS*2)
train = values[:n_train_month, :]
test = values[n_train_month:, :]

# split into input and outputs
x_train, y_train = train[:, :-1], train[:, -1]
x_val, y_val = test[:, :-1], test[:, -1]

# reshape input to be 3D [samples, timesteps, features]
x_train = x_train.reshape((x_train.shape[0], x_train.shape[1],1))
x_val = x_val.reshape((x_val.shape[0], x_val.shape[1],1))
print(x_train.shape, y_train.shape, x_val.shape, y_val.shape)
```

Fig. 3. Splitting and reshaping data

V. Metrics

- The use of Root Mean Squared Error (RMSE) is very common, and it is considered an excellent general-purpose error metric for numerical predictions [9]. RMSE is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are, then RMSE is a measure of how spread out these residuals are around the line of best fit.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(S_i - O_i)^2}$$

- The mean absolute error (MAE) is a quantity used to measure how close forecasts or predictions are to the eventual outcomes [10].

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|S_i - O_i|$$

Where in both cases O_i are the observations, S_i predicted values of a variable, and n the number of observations available for analysis.

VI. Neural Networks

Two neural networks will be considered in this project; the first one is a sequential model for LSTM neural network as on the figure on Fig.4 where the data pass from one layer to another in a sequential, the second one it is a kind of parallel LSTM memory as the second figure on Fig.4 where the the pass through four layers at the same time and the final result is concatenate in one layer.

VII. Results

In Fig.5 (t-12 t-11 ... t-1) represent 12 moths of one year and it will be the data inputs for the neural network, t represents the first month of the next next and it will be the output that the neural network will use as targets to predict.

VIII. Analysis of results

On Fig.5 the data matrix for the supervised learning problem has values between 0 and 1. That is because before preprocessing data, it has to be normalized in order to obtain a homogeneous dataset.

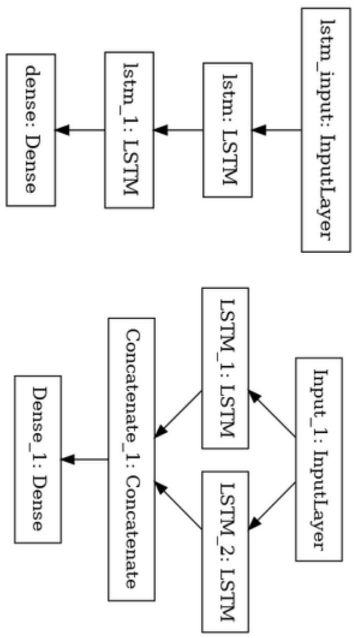
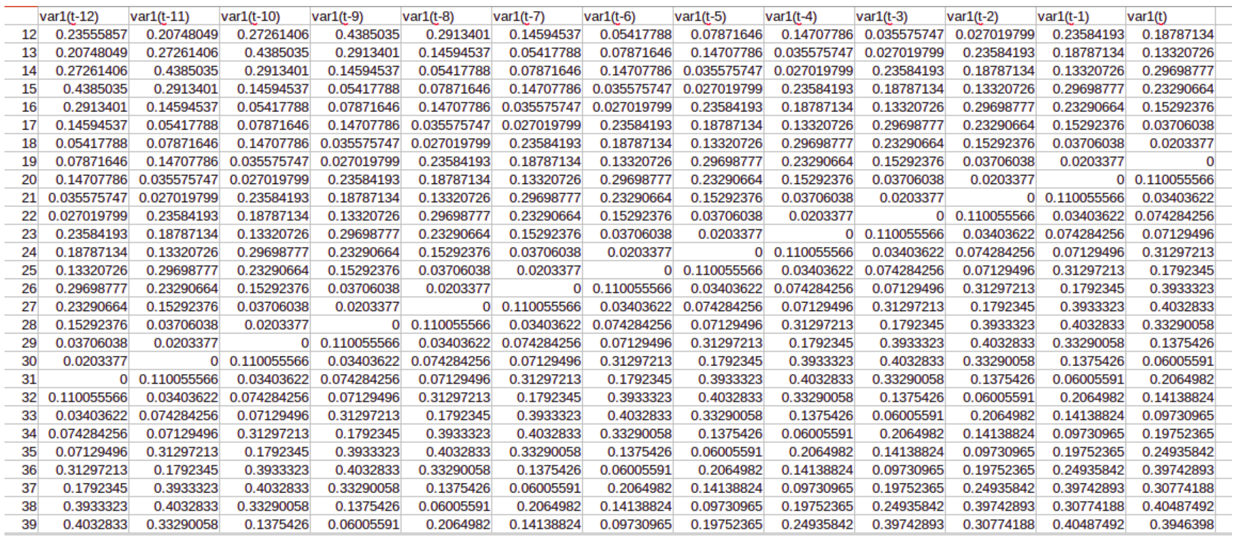


Fig. 4. Sequential and Parallel models for LSTM neural network



	var1(t-12)	var1(t-11)	var1(t-10)	var1(t-9)	var1(t-8)	var1(t-7)	var1(t-6)	var1(t-5)	var1(t-4)	var1(t-3)	var1(t-2)	var1(t-1)	var1(t)
12	0.23555857	0.20748049	0.27261406	0.4385035	0.2913401	0.14594537	0.05417788	0.07871646	0.14707786	0.035575747	0.027019799	0.23584193	0.18787134
13	0.20748049	0.27261406	0.4385035	0.2913401	0.14594537	0.05417788	0.07871646	0.14707786	0.035575747	0.027019799	0.23584193	0.18787134	0.13320726
14	0.27261406	0.4385035	0.2913401	0.14594537	0.05417788	0.07871646	0.14707786	0.035575747	0.027019799	0.23584193	0.18787134	0.13320726	0.29698777
15	0.4385035	0.2913401	0.14594537	0.05417788	0.07871646	0.14707786	0.035575747	0.027019799	0.23584193	0.18787134	0.13320726	0.29698777	0.23290664
16	0.2913401	0.14594537	0.05417788	0.07871646	0.14707786	0.035575747	0.027019799	0.23584193	0.18787134	0.13320726	0.29698777	0.23290664	0.15292376
17	0.14594537	0.05417788	0.07871646	0.14707786	0.035575747	0.027019799	0.23584193	0.18787134	0.13320726	0.29698777	0.23290664	0.15292376	0.03706038
18	0.05417788	0.07871646	0.14707786	0.035575747	0.027019799	0.23584193	0.18787134	0.13320726	0.29698777	0.23290664	0.15292376	0.03706038	0.0203377
19	0.07871646	0.14707786	0.035575747	0.027019799	0.23584193	0.18787134	0.13320726	0.29698777	0.23290664	0.15292376	0.03706038	0.0203377	0
20	0.14707786	0.035575747	0.027019799	0.23584193	0.18787134	0.13320726	0.29698777	0.23290664	0.15292376	0.03706038	0.0203377	0	0.110055566
21	0.035575747	0.027019799	0.23584193	0.18787134	0.13320726	0.29698777	0.23290664	0.15292376	0.03706038	0.0203377	0	0.110055566	0.03403622
22	0.027019799	0.23584193	0.18787134	0.13320726	0.29698777	0.23290664	0.15292376	0.03706038	0.0203377	0	0.110055566	0.03403622	0.074284256
23	0.23584193	0.18787134	0.13320726	0.29698777	0.23290664	0.15292376	0.03706038	0.0203377	0	0.110055566	0.03403622	0.074284256	0.07129496
24	0.18787134	0.13320726	0.29698777	0.23290664	0.15292376	0.03706038	0.0203377	0	0.110055566	0.03403622	0.074284256	0.07129496	0.31297213
25	0.13320726	0.29698777	0.23290664	0.15292376	0.03706038	0.0203377	0	0.110055566	0.03403622	0.074284256	0.07129496	0.31297213	0.1792345
26	0.29698777	0.23290664	0.15292376	0.03706038	0.0203377	0	0.110055566	0.03403622	0.074284256	0.07129496	0.31297213	0.1792345	0.3933323
27	0.23290664	0.15292376	0.03706038	0.0203377	0	0.110055566	0.03403622	0.074284256	0.07129496	0.31297213	0.1792345	0.3933323	0.4032833
28	0.15292376	0.03706038	0.0203377	0	0.110055566	0.03403622	0.074284256	0.07129496	0.31297213	0.1792345	0.3933323	0.4032833	0.33290058
29	0.03706038	0.0203377	0	0.110055566	0.03403622	0.074284256	0.07129496	0.31297213	0.1792345	0.3933323	0.4032833	0.33290058	0.1375426
30	0.0203377	0	0.110055566	0.03403622	0.074284256	0.07129496	0.31297213	0.1792345	0.3933323	0.4032833	0.33290058	0.1375426	0.06005591
31	0	0.110055566	0.03403622	0.074284256	0.07129496	0.31297213	0.1792345	0.3933323	0.4032833	0.33290058	0.1375426	0.06005591	0.2064982
32	0.110055566	0.03403622	0.074284256	0.07129496	0.31297213	0.1792345	0.3933323	0.4032833	0.33290058	0.1375426	0.06005591	0.2064982	0.14138824
33	0.03403622	0.074284256	0.07129496	0.31297213	0.1792345	0.3933323	0.4032833	0.33290058	0.1375426	0.06005591	0.2064982	0.14138824	0.09730965
34	0.074284256	0.07129496	0.31297213	0.1792345	0.3933323	0.4032833	0.33290058	0.1375426	0.06005591	0.2064982	0.14138824	0.09730965	0.19752365
35	0.07129496	0.31297213	0.1792345	0.3933323	0.4032833	0.33290058	0.1375426	0.06005591	0.2064982	0.14138824	0.09730965	0.19752365	0.24935842
36	0.31297213	0.1792345	0.3933323	0.4032833	0.33290058	0.1375426	0.06005591	0.2064982	0.14138824	0.09730965	0.19752365	0.24935842	0.39742893
37	0.1792345	0.3933323	0.4032833	0.33290058	0.1375426	0.06005591	0.2064982	0.14138824	0.09730965	0.19752365	0.24935842	0.39742893	0.30774188
38	0.3933323	0.4032833	0.33290058	0.1375426	0.06005591	0.2064982	0.14138824	0.09730965	0.19752365	0.24935842	0.39742893	0.30774188	0.40487492
39	0.4032833	0.33290058	0.1375426	0.06005591	0.2064982	0.14138824	0.09730965	0.19752365	0.24935842	0.39742893	0.30774188	0.40487492	0.3946398

Fig. 5. Matrix data for 12 steps in the time, it shows only the first data

IX. HPC CONSIDERATIONS

In this projects the time for loading the data is not long because it is one vector (1 x 248). However, if the vector size and window size increase, then the execution time to process all the data and convert it into a matrix could increase and it would be necessary to use some hpc techniques; such as loading data in parallel in order to decrease the execution time. Furthermore, when the neural networks are ready to work, the execution time increase because they will need to be trained, and depending on the number of epoch they will need more time and computational power.

X. CONCLUSIONS

Supervised_to_learning function and pandas library are good tools to split and reshape a data vector into a data matrix for a supervised learning problem.

REFERENCES

[1] Liu, F., & Deng, Y. (2019). A fast algorithm for network forecasting time series. IEEE Access, 7, 102554-102560.

[2] Siami-Namini, S., Tavakoli, N., & Namin, A. S. (2018, December). A comparison of ARIMA and LSTM in forecasting time series. In 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA) (pp. 1394-1401). IEEE.

[3] Sit, M., & Demir, I. (2019). Decentralized flood forecasting using deep neural networks. arXiv preprint arXiv:1902.02308.

[4] Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., & Januschowski, T. (2018). Deep state space models for time series forecasting. In Advances in neural information processing systems (pp. 7785-7794).

[5] Eckert, F., Hyndman, R. J., & Panagiotelis, A. (2020). Forecasting swiss exports using bayesian forecast reconciliation. European Journal of Operational Research.

[6] Vásquez, R. (2017). El impacto del comercio del Banano en el desarrollo del Ecuador. AFESE.

[7] Pérez, J. (2018). Estudio de factibilidad para el establecimiento de una exportadora de banano en Guayaquil, Ecuador, para su comercialización en Alemania. (Tesis de Grado). Repositorio Escuela Agrícola Panamericana Zamorano.

[8] Alvarado, P. (2020). Inversiones se reflejan en un mejor desempeño bananero. Revista Líderes

[9] Simon P. Neill, & M. Reza Hashemi.(2018) Fundamentals of Ocean Renewable Energy. E-Business Solutions

[10] Adetiloye, T., & Awasthi, A. (2017). Predicting Short-Term Congested Traffic Flow on Urban Motorway Networks. In Handbook of Neural Computation (pp. 145-165). Academic Press.