MACRO-MANAGEMENT INDICES MODELED WITH ARTIFICIAL NEURAL NETWORK. INFLUENCES ON PURCHASING POWER PARITIES

Constantin ILIE [a*]***, Margareta ILIE*** [b]

[a, b] *Ovidius University from Constanta, Romania*

ABSTRACT

The present paper shows research results regarding Artificial Neural Network simulation on macroeconomics indices and their influences over purchasing power parities. The objective was the successful build, train, and testing of different types of Artificial Neural Network to determine the best structure for applications. In modeling and simulation, authors used, not only the classical indices but also, the new ones such as online companies and houses with internet connections. The method used Artificial Neural Networks in the form of feedforward and different functions and algorithms for simulating the influence between several indices over the purchasing power parities. The results showed that's the best Artificial Neural Network hidden neurons function is tanh and the best solver four weights optimization was Adam algorithm. Applying them, the research results in absolute error smaller than 0.002 and a mean squared error smaller than $2*10^{-6}$*. These results proved that the chosen Artificial Neural Networks can be applied to the problem modeled and can be used further for a greater amount of data. The small error demonstrates the efficiency of training and the possibility of using the proposed type and features of an Artificial Neural Network.*

KEYWORDS: *Artificial Neural Network, decision making, macro management, simulation, purchasing power parities.*

DOI: 10.24818/IMC/2021/03.10

1. INTRODUCTION

Nowadays there are at least two major factors that influence our everyday lives, especially the economic activities where they show their disruptiveness. First is the COVID-19 pandemic and the second is the implementation of digitalization over the internet. The first adds uncertainty as the new online technology imposes more speed for commercial activities. So, the most important issue becomes how business management can analyze and makes decisions fast and by market changes, especially dealing with small databases or few data to work with. Also, the shift speed of economics and markets increases dramatically, and the managers need to visualize and be prepared for these modifications. This makes it important to use technologies and methods that offer the possibility to quickly make decisions to maintain de business efficiency in the market.

One of these technologies is artificial intelligence and especially the branch called Artificial Neural Networks (ANN). Not necessarily a new method, Artificial Neural Networks are used in almost all human activities, from medicine to manufacturing, from gaming to new planet colonization, from agriculture to security. Their efficiency is already demonstrated, a s their speed of development keeps pace with the need for new algorithms that offer a rapid model and simulation for different problems, systems, processes, and phenomena. In economics, at the macro- and the micro-level, the Artificial Neural Networks are used especially for forecasting, but also to determine the influences between different indices, mostly in nonlinear problems.

* Corresponding author. E-mail address: *constantin.ilie@365.univ-ovidius.ro*

Considering this, the objectives of the present work are the following: the main objective is to build an Artificial Neural Network that it's capable of models and then simulates the influence between several macroeconomic indices over the purchasing power parity index. The second objective is to see how the artificial intelligence hierarchically lists the indices according to the size of each influence, after it understands, learns, and simulates them. One of the characteristics that the present work reveals is how the Artificial Neural Networks can function with little data, knowing that, in general, artificial intelligence needs huge amounts of data to be trained and to offer an efficient result.
Considering the above, the ANN will simulate influences between indices with no direct mathematical connection and no linear impacts over each other. The importance of the research arises from the fact that the authors used not only

2. DATA

The data were collected from the European Eurostat website, and they were chosen based on the belief that these indices have a powerful influence on the purchasing power parities (PPP). The PPP was selected as the targeted one, as consumption can be disrupted by the pandemic, and the move to purchase online is increasing exponentially. The data are considered influential are production, agricultural crops, e-commerce enterprises, gross domestic product, and percentage of households with internet connection. As the first two, and, also, gross domestic product are classical indices that influence the evolution of purchasing power parities. The other two were implemented as the obvious effect of digitalization, and how the Internet impacts the way we are behaving as consumers, and how we are spending our money. The data were recorded from 2010 until 2020, and even that for some of the indices' values are exceeding this time limit, the authors were forced to remove some of the values to use only values existing in the same period.
For the use of Artificial Neural Network, we can consider that there are too few sets of data, and this can be harmful to the training process. But, considering the homogeneity of the data values and the authors' experience, we decided to proceed with the research.
The input data that we consider are:

1. Production [Volume index of production, Index, percentage changes, %, 2010=100] = Short-term business statistics (sts_inpr_a). Production in industry - annual data: mining and quarrying; manufacturing; electricity, gas, steam, and air conditioning supply; construction. For Romania between 2010-2020.
2. Crops [Area, *1000 ha)] = Permanent crops for human consumption (gov_10a_main). For Romania between 2010-2020.
3. e-commerce [Percentage of enterprises, %, All enterprises, without financial sector (10 persons employed or more)] = Enterprises with e-commerce sales (isoc_ci_eu_en2). For European Union - 27 countries between 2010-2020.
4. GDP [Total general government expenditure as percentage of gross domestic product (GDP), %] = Main revenue and expenditure items of the general government sector, notified by national authorities (gov_10a_main). For Romania between 2010-2020.
5. internet@home [Percentage of households, %] = Households with access to the internet at home (isoc_r_iacc_h). For Romania between 2010-2020.

The output data is:

6. PPP [Purchasing power parities (EU27_2020=1)] = Actual individual consumption as the exchange rates of countries' national currencies against the PPS. They express the number of currency units per PPS (prc_ppp_ind). For Romania between 2010-2020.

All input data evolutions against time years are presented in Figure 1.

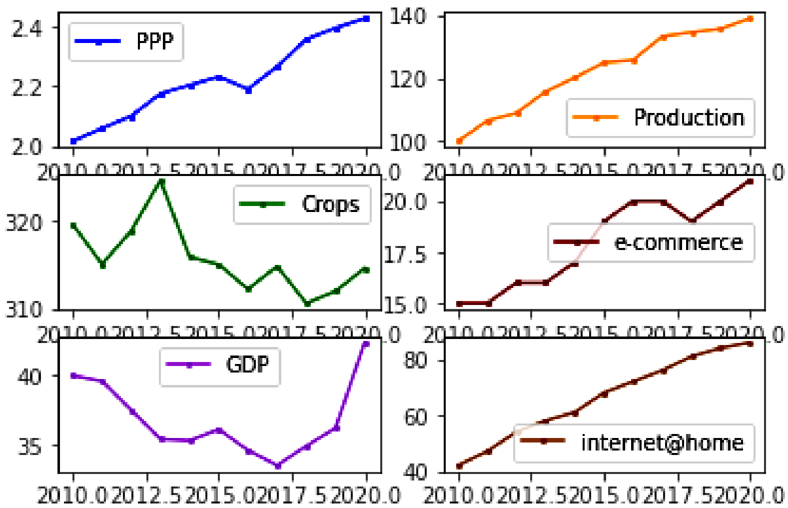


Figure 1. Indices evolution: a) purchasing power parities; b) production; c) crops; d) e-commerce; e) gross domestic product (GDP); f) houses with Internet (internet@home)
Source: Author's graphic representation using Python

As Figure 1 shows, and after a quick visual analysis of the subplots, the purchasing power parities evolution it's almost the same with production and internet@home, and, also, very similar with e-commerce. The Crops is almost inversely proportional with PPP in trend. This first analysis, just considering the visual analysis, made authors believe that the most influential indices on the partition parity power are production, e-commerce, and internet@home, while the Crops and the GDP are more heterogeneous than the other indices.

3. METHOD

As presented in the introduction, the method applied in this work is Artificial Neural Network, a bench of Artificial Intelligence. Considering the small database, the problem in hand and the experience of the authors the type of Artificial Neural Network implemented is the feed-forward. This kind of Artificial Neural Networks is the most common, easy to implement, and helpful in using different algorithms of training, numbers of hidden neurons functions, and solvers for weight optimization. In order to implement the best structure and features of the ANN two programs were used Python (using online Google Colab Notebooks) and Alyuda NeruIntelligence.
Python is open-source software and it uses a large number of modules, some of which are dedicated to Artificial Neural Network applications. Also, Python is free and can be downloaded and installed on any device (even on smartphones), and have the advantage that can be modeled with any kind of function or algorithm even that is not implemented in the program. Alyuda NeuroIntelligence is a dedicated software that has already established characteristics and structures that cannot be modified to better assess the problem but offer a very easy-to-use interface, already built in the program.
For the training algorithm, we used two types: the backpropagation algorithm for Python software and the Levenberg-Marquardt for the Alyuda NeuroIntelligence program.
The machine was a Lenovo ThinkBook with system configuration: System SKU LENOVO_MT_80EW_BU_idea_FM_Lenovo B50-80, Processor Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz, 2201 Mhz, 2 Core(s), 4 Logical Processor(s), 16 GBRAM.
Whatever the type of the neural network and its features the implementation must follow the next phases:

(a) Data Analysis.
(b) Data Preprocessing.
(c) Training.
(d) Validation and testing.
The present work is based on the steps and the results are presented in the following chapters. As brief data analysis is presented in Data chapter the following phase that will be presented is the Preprocessing one.

3.1 Preprocess

The preprocessing phase prepares the data for feeding it to the neural network to ensure an easier way for ANN to understand and learn from them. For the present problem the minmaxscaller function was applied in Python, which has the following transformation function (1):

$$x_{scalled} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

where:

$x_{scalled}$ = preprocessed value (preprocessed data value)
x = real data value.

For the Alyuda software the scaling is made with the following functions:

$$SF = \frac{SR_{max} - SR_{min}}{x_{max} - x_{min}} \quad (2)$$

$$x_p = SR_{min} + (x - x_{min}) \cdot SF \quad (3)$$

Were X - actual value of a data
X_{min} - the minimum actual value of the dataset (index)
X_{max} - the maximum actual value of the dataset (index)
SR_{min} - lower scaling range limit
SR_{max} - upper scaling range limit
SF - scaling factor
X_p - preprocessed value

The result of preprocessing scales the inputs in [-1; 1] and the outputs in [0; 1], best prepared for the neurons functions calculations.

3.2 Artificial Neural Networks feature

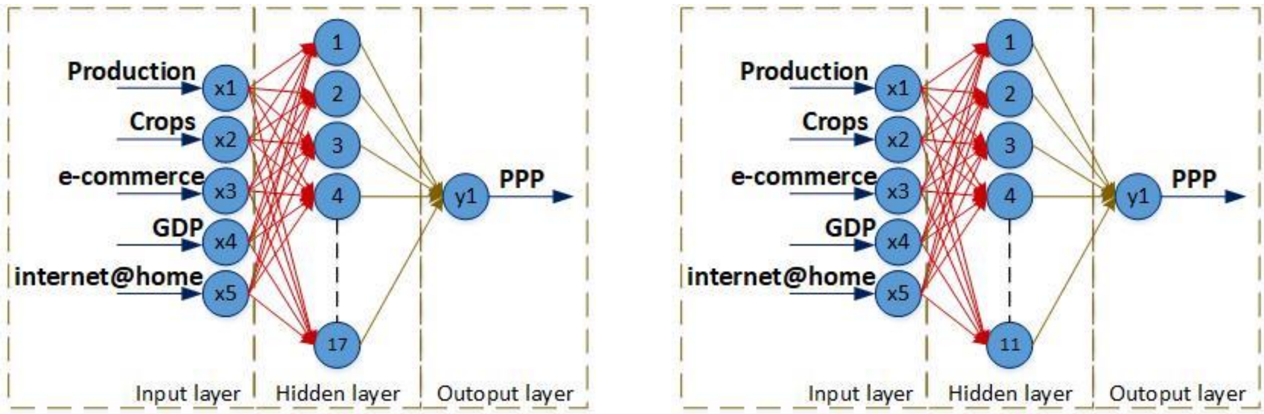
The main artificial network features are the functions from the hidden layers and the output layers, the solvers for the weight optimization, and the learning rate which is used in training. We can add other features that must consider the type of the Artificial Neural Network, the training algorithm used, etc.
For the present problem the authors choose four activation function (for the hidden layer) and two solvers for weight optimization in Python algorithm. The target was to establish the best from these functions and solvers for the further application in determining hierarchical influence. These functions and solvers are:
Type of activation function for the hidden layer and the differences between them are:

- identity - no-op activation, useful to implement linear bottleneck, returns f(x) = x.
- logistic - the logistic sigmoid function, returns f(x) = 1 / (1 + exp(-x)).
- tanh - the hyperbolic tan function, returns f(x) = tanh(x).
- relu - the rectified linear unit function, returns f(x) = max(0, x).

The solver for weight optimization:

- sgd – stochastic gradient descent.
- Adam – stochastic gradient-based optimizer proposed by Kingma, Diederik, and Jimmy Ba.

The structure of Artificial Neural Networks, as the numbers of neurons and layers depends on the problem type, numbers of datasets, and the target objective. In the present research, we searched several structures and, after comparing the results of simulations, we choose one structure for Python and one structure for Alyuda. Those structures are presented in Figure 2.



a) b)

Figure 2. Artificial Neural Network structure:
a) Alyuda: 5-17-1 neurons, b) Python, b) Alyuda.: 5-11-1 neurons

Source: Author's model using Python on Google Colab Notebooks and Alyuda NeuroIntelligence
Source: Author's structure after evaluation of several models

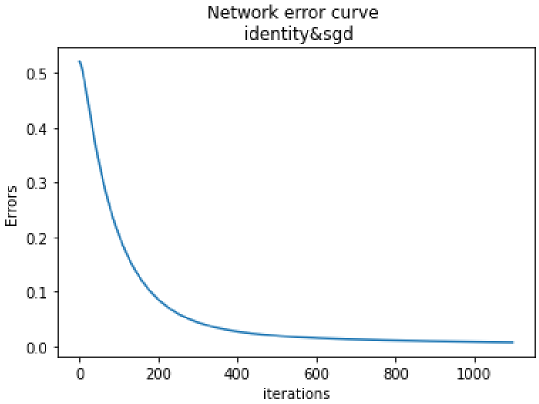
The difference between the two structures is determined especially by the type of training algorithms used (backpropagation algorithm for Python and Levenberg-Marquardt for the Alyuda NeuroIintelligence).

4. RESULTS

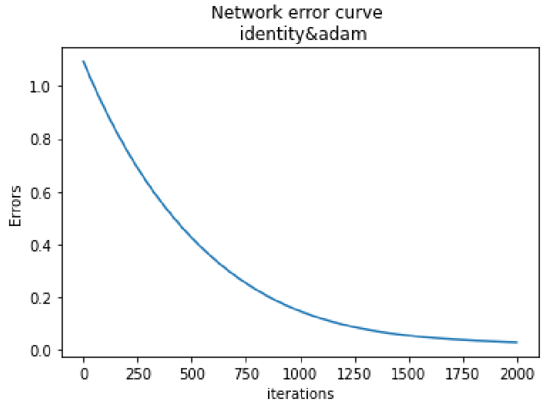
After features of Artificial Neural Network are established and set, the most important phase is the training process. After it is implemented and the results are evaluated, the Artificial Neural Network is prepared for further use for solving problems of the same type or problems with the same data kind.

4.1 Training process

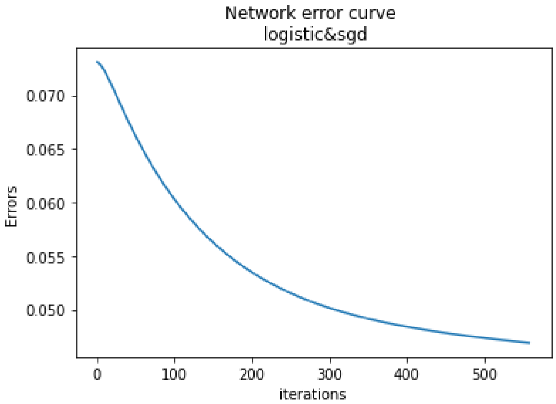
In the training process, the weights are modified at each iteration until the best-simulated outputs are calculated for the real inputs. The goal is to obtain the smallest error as the difference between the simulated outputs and the real ones, errors calculated at each iteration. The value of this difference is set by the user and it controls when the training stops as that value is reached. In the present research, the maximum difference value was set as 0.001 for Alyuda and 0.0001 for Python (the difference is, again, imposed by the training algorithm). The training process stopped when the maximum error value was reached, at 50 iterations for Alyuda and between 600 and 1400 for Python. The network error curve for each of the Artificial Neural Networks is presented in Figure 3.


Network error curve
identity&sgd
Errors
0.0
0.1
0.2
0.3
0.4
0.5
0
200
400
600
800
1000
iterations

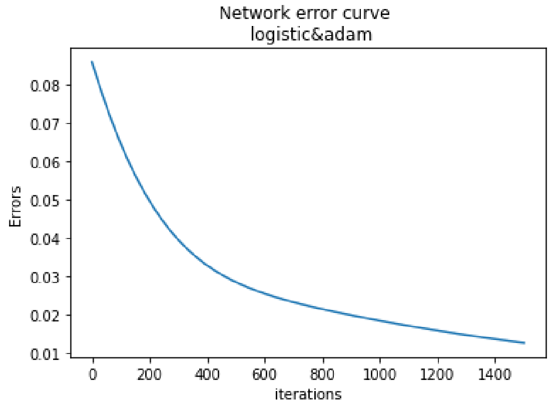
a)


Network error curve
identity&adam
Errors
0.0
0.2
0.4
0.6
0.8
1.0
0
250
500
750
1000
1250
1500
1750
2000
iterations

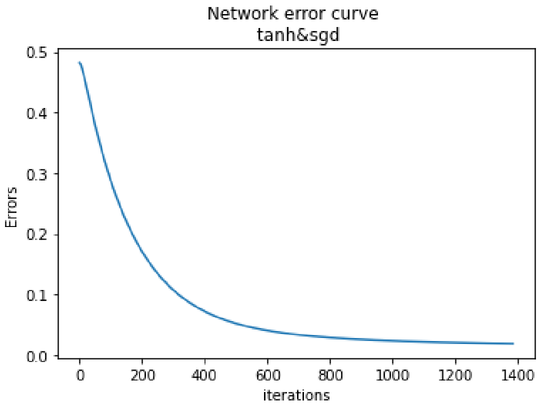
b)


Network error curve
logistic&sgd
Errors
0.050
0.055
0.060
0.065
0.070
0
100
200
300
400
500
iterations

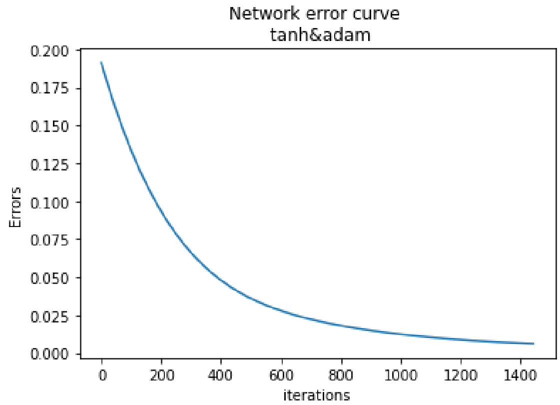
c)


Network error curve
logistic&adam
Errors
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
0
200
400
600
800
1000
1200
1400
iterations

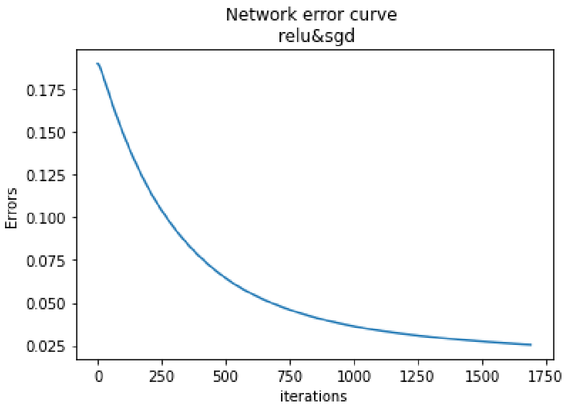
d)


Network error curve
tanh&sgd
Errors
0.0
0.1
0.2
0.3
0.4
0.5
0
200
400
600
800
1000
1200
1400
iterations

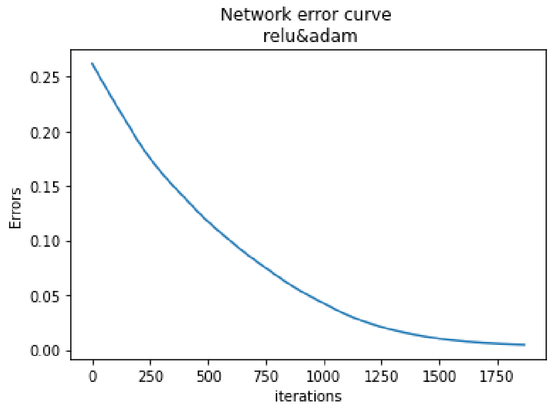
e)


Network error curve
tanh&adam
Errors
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200
0
200
400
600
800
1000
1200
1400
iterations

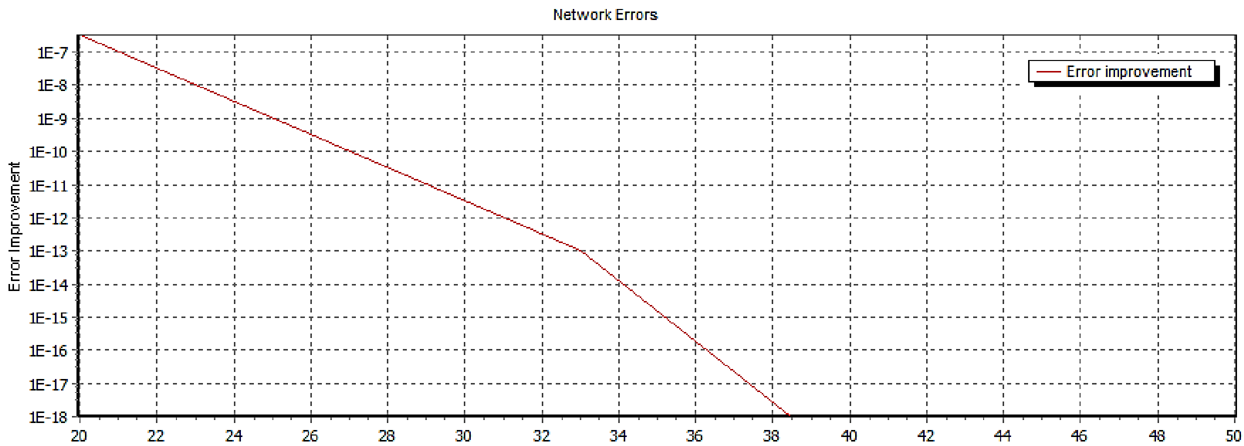
f)


Network error curve
relu&sgd
Errors
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0
250
500
750
1000
1250
1500
1750
iterations

g)


Network error curve
relu&adam
Errors
0.00
0.05
0.10
0.15
0.20
0.25
0
250
500
750
1000
1250
1500
1750
iterations

h)



i)

Figure 3. Error loss curve, algorithm: a) identity & sgd; b) identity & adam; c) logistic & sgd; d) logistic & adam; e) tanh & sgd; f) tanh & adam; g) relu & sgd; h) relu & adam; i) Levenberg-Marquardt

Source: Author's model using Python on Google Colab Notebooks and Alyuda NeuroIntelligence

The training results are also presented in Table 1 is the values of mean absolute error and mean squared error for each of the functions with a solver for weight optimization. The table contains results only from the Python program as Allyuda software was only used with hyperbolic tangent (tanh) function.

Table 1. Error evaluation with different algorithms.

Activation function for the hidden layer	Solver for weight optimization.		Solver for weight optimization.	
	sgd	adam	sgd	adam
identity	0.14994	0.07525	0.02248	0.00566
logistic	0.52919	0.15305	0.28004	0.02342
tanh	0.03437	0.00141	0.00118	1.9E-05
relu	0.02357	0.07650	0.00055	0.00585
	Mean absolute error		Mean squared error	
Alyuda NeuroIntelligence	6E-06			

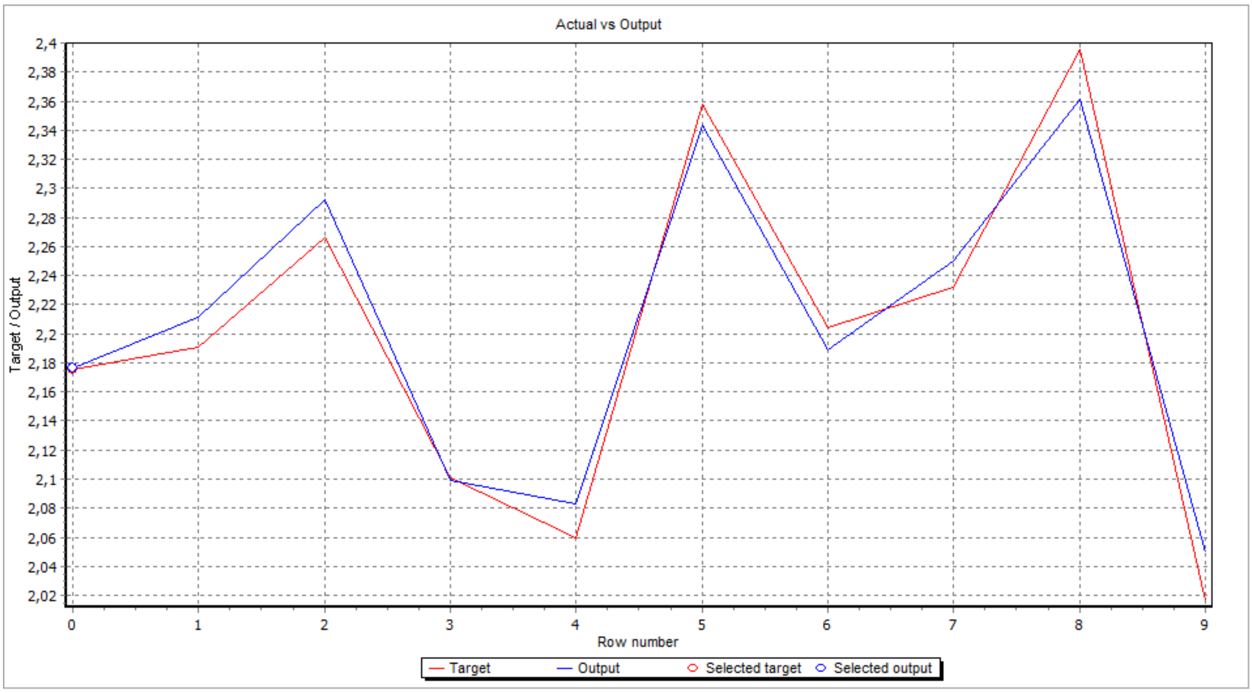
Source: Author's model using Python on Google Colab Notebooks

For an easier way to read and understand the values presented in Table 1, a color emphasis of the values was created. The smallest values are colored in green as the biggest ones are in red. As it can be seen the best, and the smallest values are gained by the use of tanh function with Adam solver. From these results, the authors further used only these two features to be implemented for the next steps in the research, like the comparison between the targets that are the real data, and the outputs as the simulated ones.

4.2 Target vs. output

The last test for the training process is the comparison between targets (real data) and the outputs data (simulated through training). For a simple comparison, 2D data can be elaborated as in Figure 4, which shows the differences between the two data values.

a)



b)

Figure 4. Comparison between targets and simulated output: a) Python's tanh & Adam logarithm; b) Alyuda NeuroIntelligence's Levenberg-Marquardt

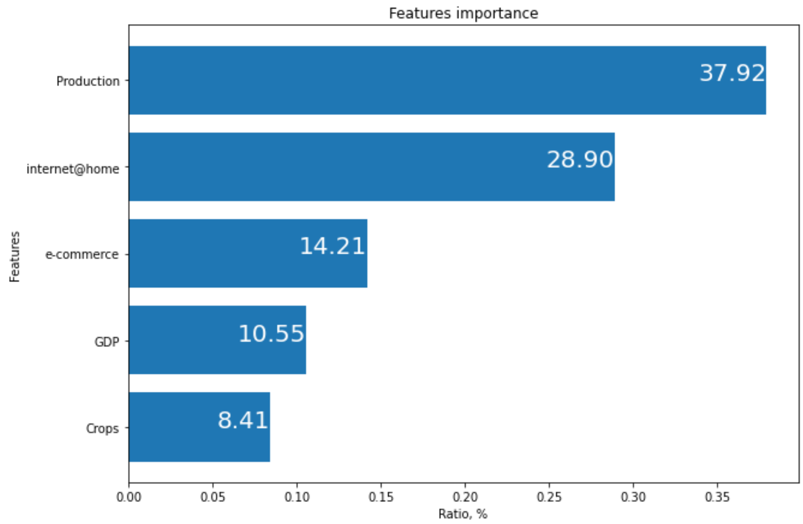
Source: Author's model using Python on Google Colab Notebooks and Alyuda NeuroIntelligence

Figure 4 shows the differences between the real data as targets, and output data as simulated ones, which are smaller than 2.5 as absolute value for Python and 0.02 as absolute values for Alyuda, for the entire training process on both programs. Such small values were considered as the evidence for the efficiency of the training and, thus, neural networks are prepared to further simulate the purchasing power parities considering new data.

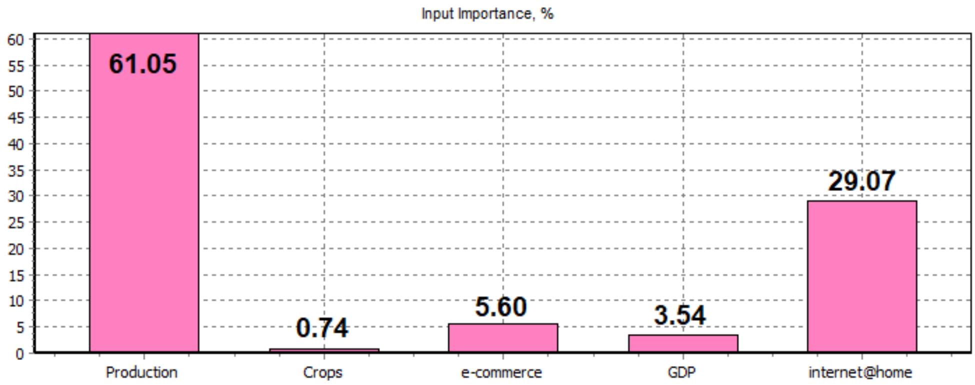
4.3 Input features importance

As one of the objectives, starting from the chosen neural networks and their features, the hierarchy of the indices influences over the outputs that were simulated are shown in figure 5.

The hierarchically listing the indices according to the size of their influence. This list shows the relative importance of each input column on the network in the training process. An input column importance is calculated as degradation in network performance after the input was removed and wasn't used by the network and can help decide what input columns can be removed with smallest effect over the training results. Thus, the more importance the ANN assigns to the input, the more it considered that the output can be calculated with it = the most important is for the output.



a)



b)

Figure 5. Feature importance [%]. a) Python's tanh & Adam logarithm; b) Alyuda NeuroIntelligence's Levenberg-Marquardt

Source: Author's model using Python on Google Colab Notebooks and Alyuda NeuroIntelligence

As can be seen in Figure 5, both programs consider that the most influential indices on the purchasing power parities are the Productions and the houses with internet, with more than 50% and even with more than 80%, when we acknowledge the Alyuda software results. Far behind there are the business involving e-commerce, then the GDP, and last the Crops. The authors will not further analyze the hierarchical list, as it is not one of the present work objectives. But we have to consider that artificial intelligence gave huge importance to the indices that involve the interaction with the internet field.

At the end of the evaluation of the training process, the authors have chosen best of the used programs. Even that Alyuda offered faster training and better errors (see subchapter 4.1 and Table 1), the authors considered that Python can get better results in future work, as it can be modeled with many several features. Also, one of Alyuda's applied training algorithms (Levenberg-Marquardt) gives the negative possibility of getting stuck in local minima, which makes the training insure.

5. CONCLUSIONS

The authors applied Artificial Intelligence techniques to simulate the influence of different macroeconomic indices. Simulating the influence of macroeconomic indices over the purchasing power parities through several Artificial Neural Networks using was a success. Comparing two types of software (open source Python and product Alyuda NeuroIntelligence) the authors chose the first one, as it was more versatile and offer more possibilities of modeling. This choice was very difficult as the Alyuda program was faster in training and offer somewhat better results in validation and testing the training results.

The comparison of several Artificial Neural Networks (in Python only) using different features results in choosing as the best neural network the one with Python's tanh function for hidden layer and Adam for the solver. With these features, the trained Artificial Neural Network simulates the influence between macroeconomics indices over the purchasing power parities with errors, calculated as differences between the output targets has real values and the simulated ones, smaller than 0.002 as absolute error and $2*10^{-5}$ as the squared error. These small error values guarantee a further efficient use of the Artificial Neural Network for simulating new sets of the same data. Also, the chosen structure of the network determines, through its simulation, a hierarchy of the influence, having on top of the list the Production index followed by the numbers of houses with internet, with a cumulated value of more than 50% of the influence. Classical indices like agricultural Crops surfaces or GDP were considered less important for the modeling of the problem and the training process.

Also, the present research demonstrates the ability of Artificial Neural Networks to understand and to be trained based on a few data, with great results simulating the nonlinear problems.

6. SCOPE FOR FUTURE WORK

The two objectives of the research were achieved as the work demonstrates the possibility of applying an Artificial Neural Network, to simulate the influence of macroeconomic indices over the purchasing power parities index. The modeled Artificial Neural Networks can be used for simulating different kinds of influences between macro- and micro-economic indices. Also, Artificial Neural Networks can offer new ways of showing how the new technologies can be viewed, acknowledging the more influence they have, and the more classical ones are overcome and minimized by the social and economic evolutions.

This should be considered for future work, where more indices should be involved, and also the size of the data set should be increased. Other features can be implemented to respond faster and better to the quick change or shifts of the market, due to nonlinear incidents like financial crises, pandemics, etc.

REFERENCES

European Commission Eurostat, *Short-term business statistics – [sts_inpr_a]*, Retrieved September 10, 2021, from https://ec.europa.eu/eurostat/cache/metadata/en/sts_esms.htm#unit_measure1626422794870

European Commission Eurostat, *Crop production – [apro_cpnh1]*, Retrieved September 10, 2021, from https://ec.europa.eu/eurostat/cache/metadata/en/apro_cp_esms.htm

European Commission Eurostat, *Summary of EU aggregates – [isoc_ci_eu_en2]*, Retrieved September 10, 2021, from https://appsso.eurostat.ec.europa.eu/nui/submitViewTableAction.do

European Commission Eurostat, *Government revenue, expenditure and main aggregates – [gov_10a_main]*, Retrieved September 10, 2021, from https://ec.europa.eu/eurostat/cache/metadata/en/gov_10a_main_esms.htm

European Commission Eurostat, *ICT usage in households and by individuals (isoc_i) - [isoc_r_iacc_h]*, Retrieved September 10, 2021, from https://ec.europa.eu/eurostat/cache/metadata/en/isoc_i_esms.htm#stat_pres1625659541939

European Commission Eurostat, *Purchasing power parities - [prc_ppp_ind]*, Retrieved September 10, 2021, from https://ec.europa.eu/eurostat/cache/metadata/en/prc_ppp_esms.htm