Universitat Rovira i Virgili

Delivery of Activity 1

Neural and Evolutionary Computation

Prediction with Back-Propagation and Linear Regression

Javier Novella Ruiz

Contents

1 Introduction

The goal of this activity is to practice with different types of supervised models. In particular, the following methods have been used.

- Neural Network with Back-Propagation BP, implemented by the author of this report.
- Neural Network with Back-Propagation BP-F, using keras library.
- Multiple Linear Regression MLR-F, using the sklearn python library.

To achieve this purpose firstly the a dataset has been selected, analyzed and pre-processed. Then the methods commented before have been applied to the dataset to obtain predictions of values.

1.1 GitHub repository

The Jupyter Notebook and the module with the BP devolpemnt together with the dataset files and image results can be consulted in the following GitHub repository (Link to Javier Novella NEC A1 repository).

From the descriptions and explanations point of view, the Jupyter Notebook file A1.ipynb contains a retailed version of this report in the markdowns. But it contains all the Python code used to obtain the results for this report.

For the BP implementation the file BP.py is the module which implements the NeuralNet Class, and can be found also in the same repository.

2 Dataset

The dataset used include data for the estimation of obesity levels in individuals from the countries of Mexico, Peru and Colombia, based on their eating habits and physical condition. The data contains 17 attributes and 2111 records. The dataset source is UCI Machine Learning Repository (Link to the dataset repository).

Note: all the results and calculations obtained in this chapter can be found in the file A1.ipynb in the GitHub repository (Link to A1.ipynb in Javier Novella NEC A1 repository)

The dataset is composed by the following features:

- **Gender**: gender of the individual.
- **Age**: age of the individual.
- **Height**: height of the individual.
- **Weight**: weight of the individual.
- **family_history_with_overweight**: Has a family member suffered or suffers from overweight?
- **FAVC**: Does the individual eat high caloric food frequently?
- **FCVC**: Does the individual usually eat vegetables in your meals?
- **NCP**: How many main meals does the individual have daily?
- **CAEC**: Does the individual eat any food between meals?
- **SMOKE**: Does the individual smoke?
- **CH20**: How much water does the individual drink daily?
- **SCC**: Does the individual monitor the calories eaten daily?
- **FAF**: How often the individual has physical activity?
- **TUE**: How much time does the individual use technological devices such as cell phone, videogames, television, computer and others?
- **CALC**: How often does the individual drink alcohol?
- **MTRANS**: Which transportation does the individual usually use?
- **NObeyesdad**: Obesity level

2.1 Inputs and output selection

From the 17 features, 16 of them are considered inputs and 1 is considered output.

- **Inputs:**
  - Gender
  - Age
  - Height
  - Weight
  - family_history_with_overweight
  - FAVC
  - FCVC
  - NCP
  - CAEC
  - SMOKE
  - CH20
  - SCC
  - FAF
  - TUE
  - CALC
  - MTRANS
- **Outputs:**
  - NObeyesdad

2.2 Output transformation

The output NObeyesdad allows classification of the data using the values of Insufficient Weight, Normal Weight, Overweight Level I, Overweight Level II, Obesity Type I, Obesity Type II and Obesity Type III. As the predicted variable must be a real value $floatordouble$ this value will be replaced by the corresponding BMI.

The Body Mass Index BMI is a simple calculation using a person's height and weight. The formula is $BMI = \frac{weight}{height^2}$ $(\frac{kg}{m^2})$. It is used as an indicator of whether an individual is underweight, normal weight, overweight, or obese. The correspondance from the previous categorical values to the new numerical ones are:

- **Insufficient Weight (Underweight)**: BMI less than 18.5
- **Normal Weight**: BMI between 18.5 and 24.9
- **Overweight Level I (Pre-obesity)**: BMI between 25 and 29.9
- **Overweight Level II**: BMI between 30 and 34.9
- **Obesity Type I**: BMI between 35 and 39.9

- **Obesity Type II**: BMI between 40 and 44.9
- **Obesity Type III**: BMI 45 and above

To achive this result, the value of the column "NObeyesdad" is replaced by the mentioned formula $BMI = \frac{weight}{height^2}$ using the weight and height within the same record processed. This will result in the BMI calcualted for each individual which is a real numeric value.

2.3 Dataset checks

The data will be checked for the following points:

- Check for missing values
- Check for duplicates
- Check data type
- Check unique values on each column

2.3.1 Check missing values

The dataset is fully checked for null values feature by feature. No missing values are found in during this search.

Feature	Number of missing values
Gender	0
Age	0
Height	0
Weight	0
family_history_with_overweight	0
FAVC	0
FCVC	0
NCP	0
CAEC	0
SMOKE	0
CH2O	0
SCC	0
FAF	0
TUE	0
CALC	0
MTRANS	0
Nobeyesdad	0

Table 1: Number of missing values within the full dataset

2.3.2 Check duplicates

24 records are duplicated. This means that the dataset contains information about 24 individuals that have exactly the same features. In this case, this duplicates are kept because it is assumed that they represent legitimate repeated records from different people.

2.3.3 Check datatypes

Check data types allows to make a first sight to know which features are numerical and which are categorical. This will help in the further pre-processing steps to know which fields must be transformed/normalized. In this case the features are:

Feature	Data type	Type
Gender	object	Categorical
Age	float64	Numerical
Height	float64	Numerical
Weight	float64	Numerical
family_history_with_overweight	object	Categorical
FAVC	object	Categorical
FCVC	float64	Numerical
NCP	float64	Numerical
CAEC	object	Categorical
SMOKE	object	Categorical
CH2O	float64	Numerical
SCC	object	Categorical
FAF	float64	Numerical
TUE	float64	Numerical
CALC	object	Categorical
MTRANS	object	Categorical
NObeyesdad	float64	Numerical

Table 2: Data types of each feature

2.3.4 Check unique values by feature

Understanding the range of unique values in categorical variables will help in choosing the right encoding techniques.

Feature	Number of unqiue values
Gender	2
Age	1402
Height	1574
Weight	1525
family_history_with_overweight	2
FAVC	2
FCVC	810
NCP	635
CAEC	4
SMOKE	2
CH2O	1268
SCC	2
FAF	1190
TUE	1129
CALC	4
MTRANS	5
Nobeyesdad	1340

Table 3: Number of unique values by feature

2.4 Statistics

The dataset consists of 2111 records and 17 features, with each record considered unique despite some duplications. Importantly, there are no null values in the dataset, ensuring completeness. The features cover aspects of individuals' health and lifestyle, including height, weight, vegetable intake, number of main meals, water consumption, sports activity, use of technological devices, and BMI.

	Age	Height	Weight	FCVC	NCP	CH2O	FAF	TUE	NObeyesdad
count	2111	2111	2111	2111	2111	2111	2111	2111	2111
mean	24.310	1.700	86.590	2.420	2.690	2.010	1.010	0.660	29.700
std	6.350	0.090	26.190	0.530	0.780	0.610	0.850	0.610	8.010
min	14.000	1.450	39.000	1.000	1.000	1.000	0.000	0.000	13.000
25%	19.950	1.630	65.470	2.000	2.660	1.580	0.120	0.000	24.330
50%	22.780	1.700	83.000	2.390	3.000	2.000	1.000	0.630	28.720
75%	26.000	1.770	107.430	3.000	3.000	2.480	1.670	1.000	36.020
max	61.000	1.980	173.000	3.000	4.000	3.000	3.000	2.000	50.810

Table 4: Statistics from the dataset

The dataset shows a range of values for these features. Heights range from 1.45 to 1.98 meters, and weights range from 39.0 to 173.0 kilograms. Individuals consume between 1 to 3 pieces of vegetables per day and have 1 to 4 main meals daily. Water intake varies from 1 to 3 liters per day, sports activity ranges from 0 to 3 hours per day, and use of technological devices ranges from 0 to 2 hours per day. BMI values span from 13.0 to 50.81 kg/m^2, indicating a wide range of body compositions.

In terms of data distribution, the features with the highest variability are age, weight, and

obesity level. The averages of these features are close to the 50th percentile, suggesting that the data is evenly distributed. This balanced distribution is beneficial for analysis, as it indicates that the dataset provides a comprehensive representation of the population's health and lifestyle characteristics.

2.4.1 Categorical features

The categorical features and its exposed in the Table 5.

Feature	Possible values
Gender	['Female' 'Male']
family_history_with_overweight	['yes' 'no']
FAVC	['no' 'yes']
CAEC	['Sometimes' 'Frequently' 'Always' 'no']
SMOKE	['no' 'yes']
SCC	['no' 'yes']
CALC	['no' 'Sometimes' 'Frequently' 'Always']
MTRANS	['Public_Transportation' 'Walking' 'Automobile', 'Motorbike' 'Bike']

Table 5: Possible values of the categorical features

In the Figure 1 it shown the distribution of the categorical data. Showing the distribution is important when analyzing a dataset for neural network usage because it helps identify imbalances and patterns within the data. This understanding can help training on a representative sample, and this can improve its accuracy and generalization. It also shows any potential biases, allowing for adjustments to be made to achieve good predictions.

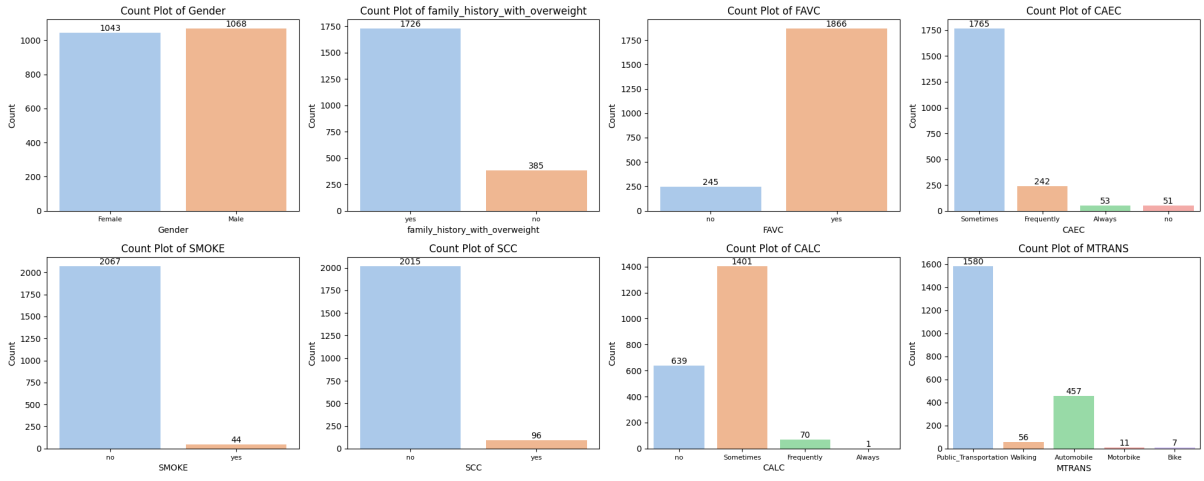

Figure 1: Distribution of the categorical features

But the get more insights it is interesting to get the boxplot of the categorical features against the output feature that is the obesity level. This is shown in the Figure 2. This view provide a clear summary of the distribution, central tendency, and variability of obesity levels within each category. This helps identify patterns, outliers, and differences between groups, making it easier to understand how different categorical factors influence obesity.

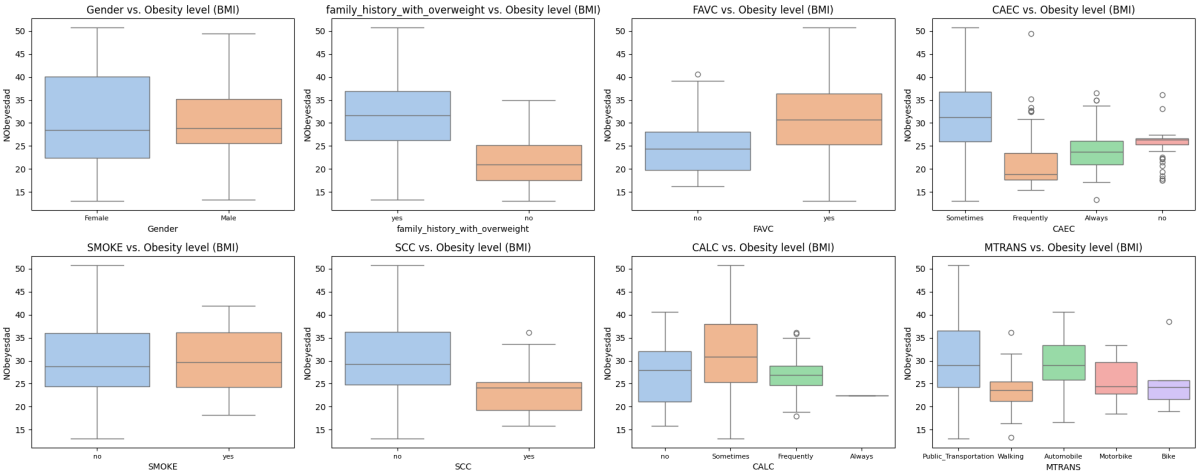

Figure 2: Categorical features vs obesity level

From this data, can be exposed that males tend to have higher levels of obesity compared to females. High caloric diets are associated with increased obesity levels, while controlling calorie intake helps in maintaining lower obesity levels. Interestingly, smoking does not seem to impact obesity levels significantly.

Additionally, individuals who use active transportation methods like walking or biking tend to have lower obesity levels. These findings highlight the importance of diet and physical activity in managing obesity.

2.4.2 Numerical features

In the Figure 3 it shown the distribution of the numerical data. Showing the distribution of numerical features using histograms is important because it helps visualize the data's shape, central tendency, and spread.

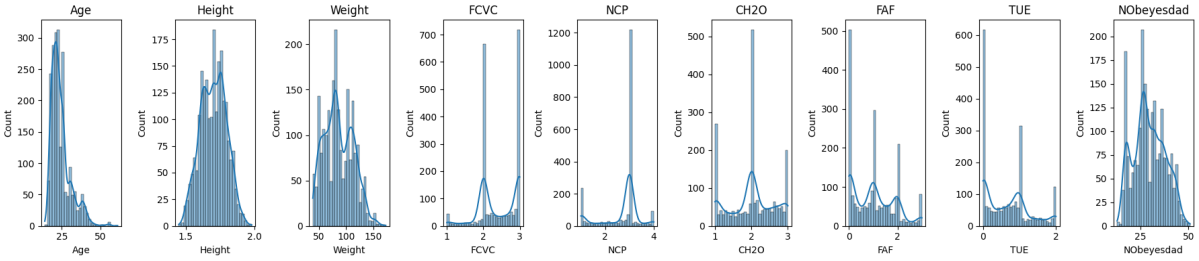

Figure 3: Distribution of the numerical features

It is interesting to focus on the output variable, to get more insights from the dataset, this is shown in the Figure 4. Focusing on the distribution of the obesity level, is crucial because it helps understand the range and variability of the variable.

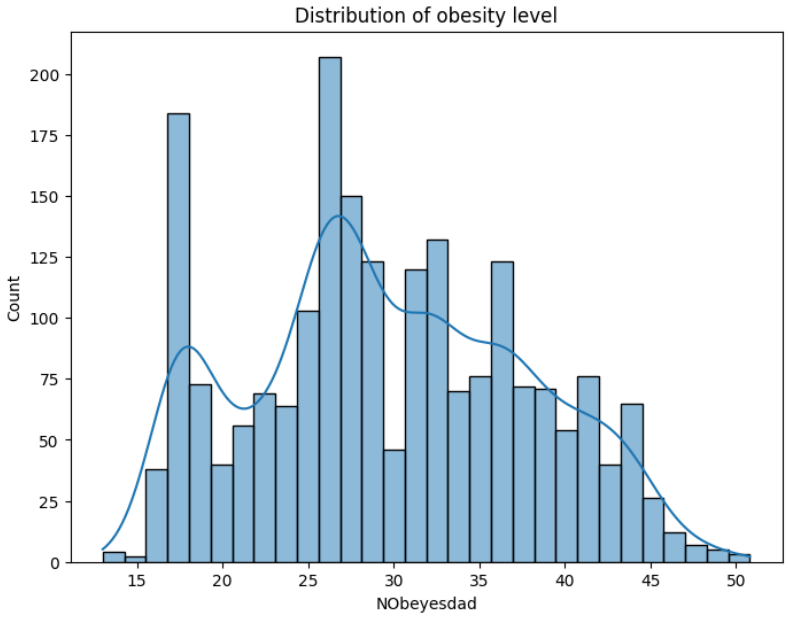

Figure 4: Distribution of the obesity level

The relation between this numerical data and its affectance to the output variable can be seen in the correlation matrix shown in the Figure 5. A heatmap of the correlation matrix is used because it visually represents the strength and direction of relationships between the numerical features. This helps identify which features are strongly correlated with each other and also with the obesity level.

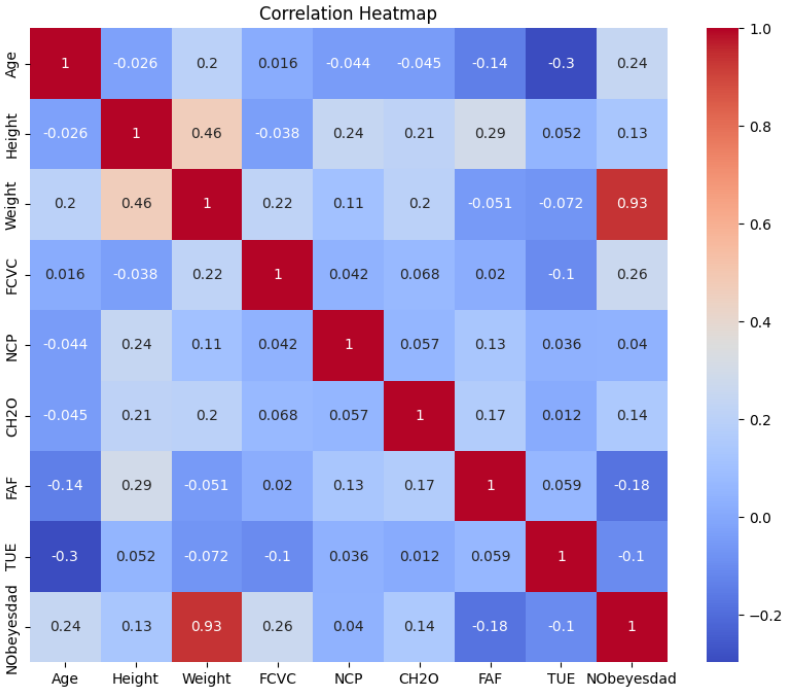

Figure 5: Coorrelation matrix between numerical features

From this data is deduced taht the individuals in the dataset have a BMI ranging from 13.0 to

50.81 kg/m². The average obesity level is 29.7, which corresponds to Overweight Level I, close to the threshold for Overweight Level II. It is important to note that there is high variability in obesity levels, with 25% of individuals having normal BMI values ($BMI < 25$).
Moreover the features with the highest impact on obesity levels, in decreasing order, are Weight, FCVC: frequency of consumption of vegetables, Age, CH20: water consumption, and Height.

2.5 Data preprocessing

The data is splitted in two different datasets, the training and test datasets. The training dataset contains the 16 input features stated before while the test data set only contains the output fetaure NObeyesdad=Obesity level. The datasets are created from the main dataset with 2111 records and it is splitted such 80% of the records (1688) are in the train set and the 20% remaining (423) are in the test set. From this train set the 75% of the records is kept as training set (1583) and the 25% of the records (528) is used as validation data.

The following categorical features has been adjusted for integer values, this is suitable because the categorical features in the dataset have an inherent order (low, medium, high etc.) or the features are binary which can be converted from the 2 binary values to 0 and 1, which is a straightforward and effective change.

Feature	**Type**
Gender	Categorical
family_history_with_overweight	Categorical
FAVC	Categorical
CAEC	Categorical
SMOKE	Categorical
SCC	Categorical
CALC	Categorical
MTRANS	Categorical

Table 6: Categorical features preprocessed

The following numerical values has been normalized. This is important to improve convergence speed and accuracy, preventing feature dominance and improving interpretability.

Feature	**Type**
Age	Numerical
Height	Numerical
Weight	Numerical
FCVC	Numerical
NCP	Numerical
CH2O	Numerical
FAF	Numerical
TUE	Numerical
NObeyesdad	Numerical

Table 7: Numerical features preprocessed

3 Implementation of BP

The BP Python module includes activation functions and the NeuralNet class for implementing neural networks with backpropagation. The module provides several activation functions: Sigmoid, ReLU, Leaky ReLU, Tanh, and Linear. It has attributes like network structure, epochs, learning rate, and momentum. The activation function can be selected using the fact parameter.

Note: this implementation can be found in the file BP.py in the GitHub repository (Link to BP.py in Javier Novella NEC A1 repository)

3.1 NeuralNet class

3.1.1 Attributes

The list of attributes and their descriptions are the following ones:

- **Network Structure**: The layers attribute specifies the number of neurons in each layer.
- **Epochs**: The epochs attribute defines the number of training iterations.
- **Learning Rate**: The learning_rate attribute controls the step size during gradient descent.
- **Momentum**: The momentum attribute helps accelerate gradients vectors in the right directions.
- **Activation Function**: The fact attribute allows selecting the activation function to be used in the network.

3.1.2 Private methods

The **_select_activation_function()** method dynamically selects the appropriate activation function based on the fact parameter. It supports functions like sigmoid, relu, leaky relu, tanh, and linear.

The **_select_derivative_activation_function()** method selects the derivative of the activation function, which is for backpropagation. It matches the activation function chosen by _select_activation_function().

The **_forward_propagation()** method passes input data through the network layer by layer. It applies weights, thresholds, and activation functions to compute the output of each layer.

The **_backward_propagation()** method calculates the output error and propagates it backward through the network. It updates weights and thresholds using gradient descent with momentum to minimize the error.

The **_train()** method normalizes input data and trains the network using mini-batch gradient descent. It periodically computes the training and validation loss to monitor the network's performance.

3.1.3 Public methods

The **fit()** method normalizes input data and trains the network. It calls the _train() method to perform the training process.

The **predict()** method normalizes input data and performs a forward pass through the network to generate predictions on new data.

The **loss_epochs()** method returns the training and validation loss for each epoch.

3.2 Activation functions

3.2.1 Sigmoid

The sigmoid activation function is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Its derivative is:

$$\sigma'(x) = \sigma(x) \cdot (1 - \sigma(x))$$

3.2.2 ReLU

On the one hand, the ReLU (Rectified Linear Unit) activation function is defined as:

$$\text{ReLU}(x) = \max(0, x)$$

Its derivative is:

$$\text{ReLU}'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

On the other, the Leaky ReLU activation function is defined as:

$$\text{Leaky ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases}$$

Its derivative is:

$$\text{Leaky ReLU}'(x) = \begin{cases} 1 & \text{if } x > 0 \\ \alpha & \text{if } x \leq 0 \end{cases}$$

3.2.3 Linear

The linear activation function is defined as:

$$f(x) = x$$

Its derivative is:

$$f'(x) = 1$$

3.2.4 Tanh

The tanh (hyperbolic tangent) activation function is defined as:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Its derivative is:

$$\tanh'(x) = 1 - \tanh^2(x)$$

4 Testing the implemented BP model

4.1 Results

Test nº	Nº layers	Structure	Nº epochs	Learning	Momentum	Activation
1	3	[16, 4, 1]	10000	0.001	0.001	tanh
2	4	[16, 8, 4, 1]	10000	0.001	0.001	relu
3	5	[16, 8, 4, 2, 1]	10000	0.01	0.01	linear
4	3	[16, 4, 1]	1000	0.001	0.001	tanh
5	3	[16, 4, 1]	1000	0.001	0.001	linear

Table 8: Parameters used for the tests done to the BP implementation

Test nº	MAPE	MAE	MSE
1	3.745677e+11	5.346505e-2	3.229095e-3
2	6.461173e+11	4.253340e-2	2.032964e-3
3	2.684181e+11	7.270744e-2	5.894476e-3
4	5.895478e+11	6.132290e-2	4.673645e-3
5	3.589370e+11	4.678564e-2	2.941892e-3

Table 9: Results of the tests done to the BP implementation

4.2 Analysis of the parameters used

4.2.1 Learning rate and momentum

Tests showed that higher learning rates and momentum significantly worsened results. The table below shows one test where errors increased tenfold with higher parameters. Therefore, it's best to keep the learning rate and momentum low.

Nº layers	Structure	Nº epochs	Learning	Momentum	Activation
3	[16, 4, 1]	10000	0.1	0.1	tanh

Table 10: Parameters used for the test with higher learning rate and momentum

MAPE	MAE	MSE
4.926861e+11	1.344565e-1	2.941442e-2

Table 11: Results of the test with higher learning rate and momentum

4.2.2 Activation function

From the activation function perspective, linear and tanh tend to yield better results. Under the same conditions, linear shows lower MAPE and MAE, while tanh has a lower MSE.

4.3 Loss

With 3 layers, a learning rate of 0.001, and the tanh activation function, Test 1 achieved the lowest MAE and MSE, indicating good performance. Comparing it with Test 4, which uses the same parameters but fewer epochs, shows better performance in Test 1. This suggests that a

higher number of epochs tends to reduce the error. The training and validation loss plots show that after some epochs, the values stabilize, and the reduction per epoch decreases, but overall, the loss continues to decrease.

From Tests 1, 2, and 3, the training and validation errors are plotted to show their evolution over the epochs.

4.3.1 Test 1

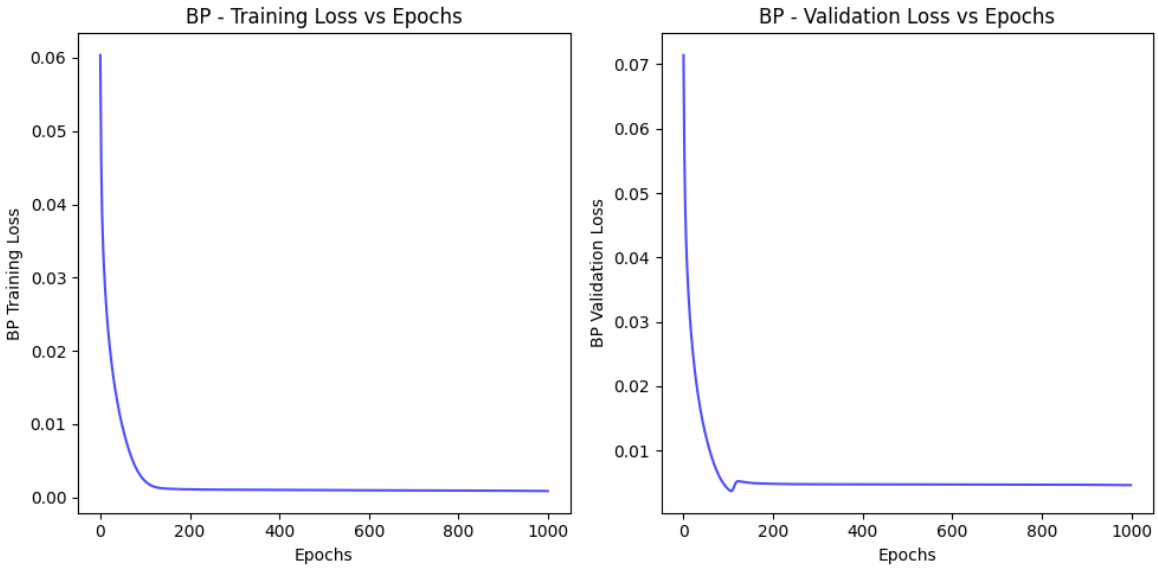

Figure 6: Training error vs validation error in test index 1

4.3.2 Test 2

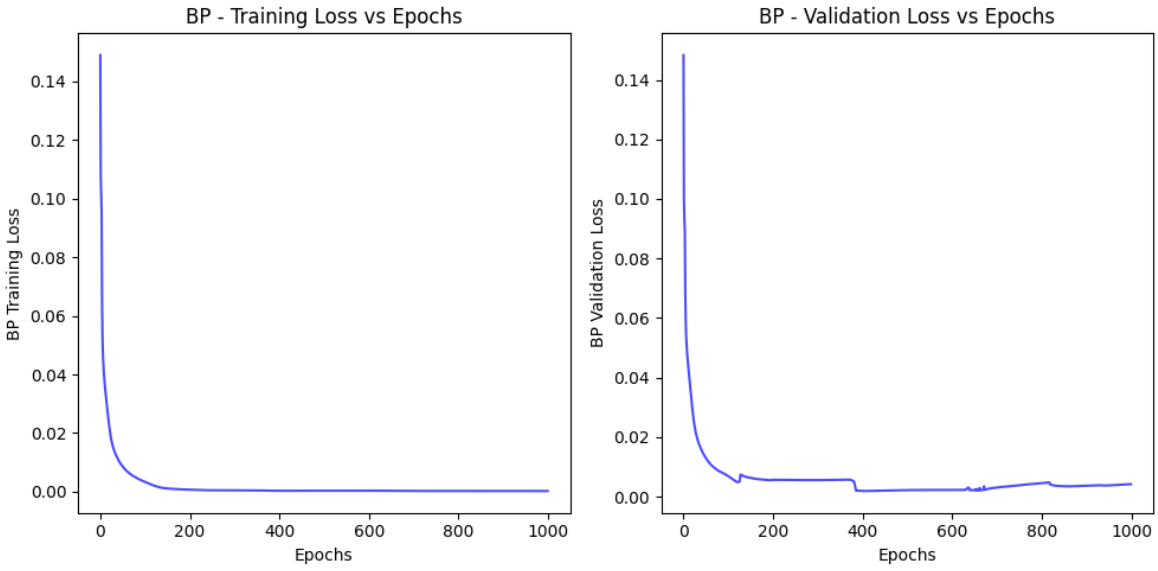

Figure 7: Training error vs validation error in test index 2

4.3.3 Test 3

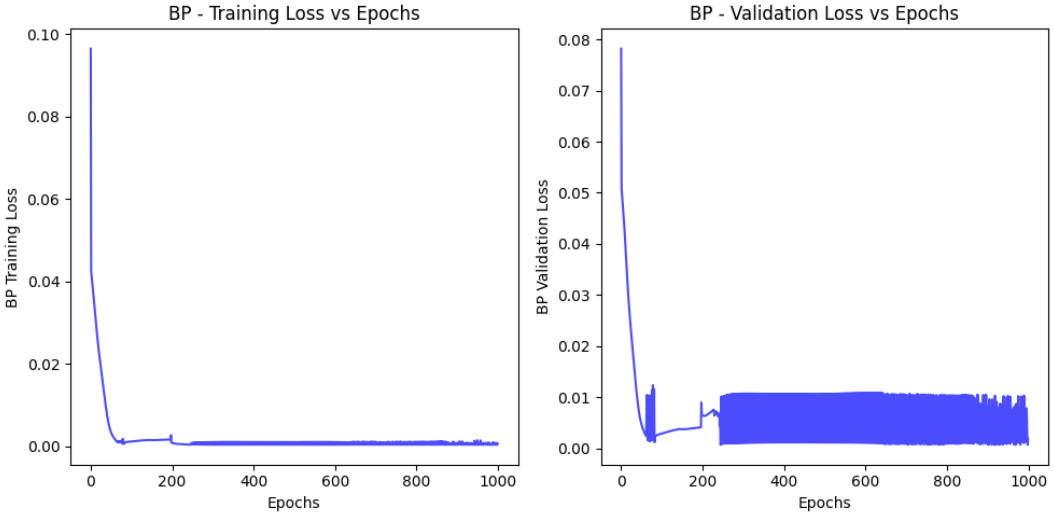

Figure 8: Training error vs validation error in test index 3

5 Comparison with other models BP vs BP-F vs MLR-F

When evaluating model performance, scatter plots are a useful tool. In these plots, better performance is indicated by points closer to the diagonal. This means the predicted values are closer to the real ones. Three new tests have been performed to compare the three models.

From the model's perspective, shows some clear trends. In Test 1, the BP-F model outperformed both BP and MLR-R models, achieving the lowest MAPE, MAE, and MSE. This indicates that BP-F is more accurate and reliable under these conditions.

In Test 2, BP-F again showed better performance than BP and MLR-R, with lower MAE and MSE values. However, the difference between BP and BP-F was less pronounced compared to Test 1. This suggests that while BP-F generally performs better, the advantage may vary depending on the specific test conditions.

Test 3 presented a different scenario where BP achieved the lowest errors, while BP-F and MLR-R had significantly higher MAPE, MAE, and MSE values. This indicates that BP can sometimes outperform BP-F and MLR-R, especially when the number of epochs is lower.

In conclusion and overall, BP and BP-F models tend to perform better than MLR-R, but the choice between BP and BP-F may depend on the specific parameters and conditions of the test.

5.1 Test 1

The parameters used for the 3 models are the following ones:

Model	**Nº layers**	**Structure**	**Nº epochs**	**Learning**	**Momentum**	**Activation**
BP	3	[16, 4, 1]	10000	0.001	0.001	relu
BP-F	3	[16, 4, 1]	10000	0.001	0.001	relu
MLR-F	3	[16, 4, 1]	10000	0.001	0.001	relu

Table 12: Test 1 - Comparison between models parameters

And the quality results are the following ones

Model	**MAPE**	**MAE**	**MSE**
BP	8.174308e+11	8.011910e-2	6.945218e-3
BP-F	5.832255e+11	5.275115e-2	3.506288e-3
MLR-F	1.812658e+12	8.446904e-2	1.219164e-2

Table 13: Test 1 -Comparison between models results

5.1.1 Scatter plot

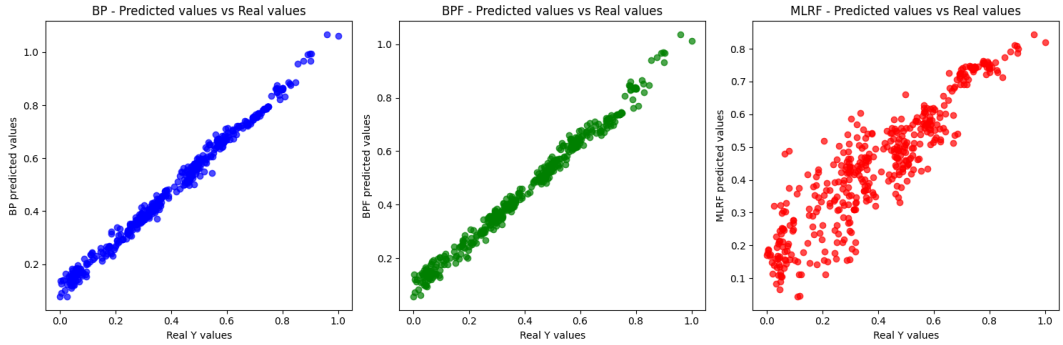

Figure 9: Test 1 - scatter plot of the 3 models

5.2 Test 2

The parameters used for the 3 models are the following ones:

Model	Nº layers	Structure	Nº epochs	Learning	Momentum	Activation
BP	3	[16, 4, 1]	1000	0.001	0.001	relu
BP-F	3	[16, 4, 1]	1000	0.001	0.001	relu
MLR-F	3	[16, 4, 1]	1000	0.001	0.001	relu

Table 14: Test 2 - Comparison between models parameters

And the quality results are the following ones

Model	MAPE	MAE	MSE
BP	6.239873e+11	7.014470e-2	5.567292e-3
BP-F	6.572347e+11	5.234516e-2	3.462514e-3
MLR-F	1.568601e+12	6.542944e-2	7.089106e-3

Table 15: Test 2 -Comparison between models results

5.2.1 Scatter plot

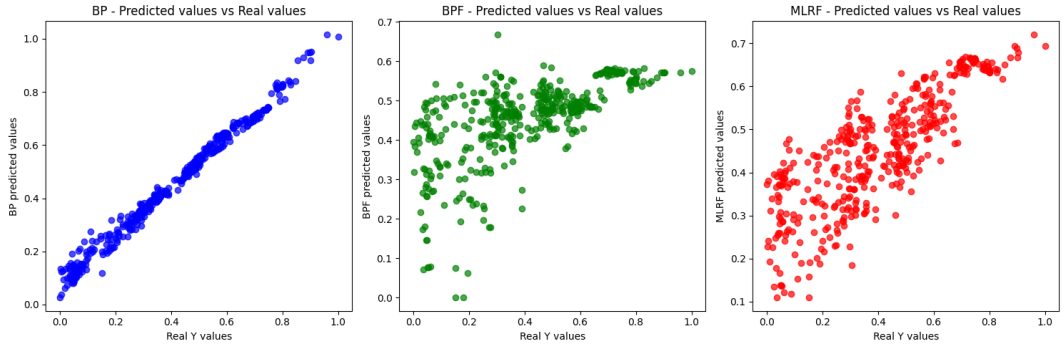

Figure 10: Test 2 - scatter plot of the 3 models

5.3 Test 3

The parameters used for the 3 models are the following ones:

Model	Nº layers	Structure	Nº epochs	Learning	Momentum	Activation
BP	3	[16, 4, 1]	1000	0.001	0.001	relu
BP-F	3	[16, 4, 1]	1000	0.001	0.001	relu
MLR-F	3	[16, 4, 1]	1000	0.001	0.001	relu

Table 16: Test 3 - Comparison between models parameters

And the quality results are the following ones

Model	MAPE	MAE	MSE
BP	2.764482e+11	4.838344e-2	3.096494e-3
BP-F	4.212754e+12	1.542775e-1	3.621248e-2
MLR-F	3.964180e+12	1.188944e-1	2.303858e-2

Table 17: Test 3 -Comparison between models results

5.3.1 Scatter plot

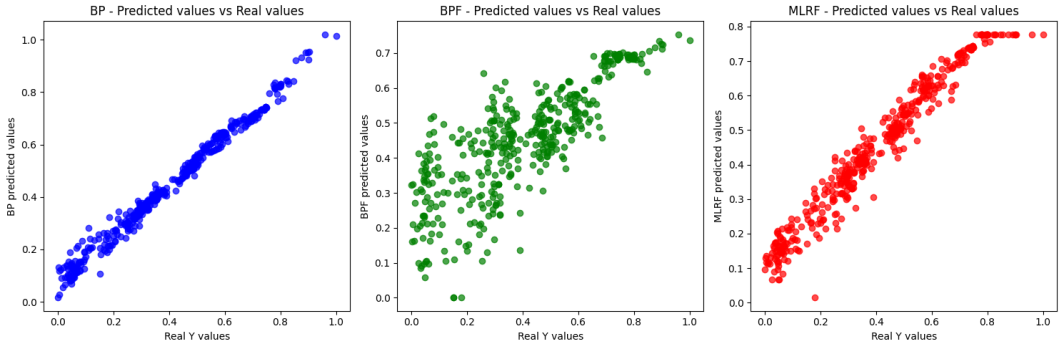

Figure 11: Test 3 - scatter plot of the 3 models

References

[1] Sergio Gómez, "Supervised Learning Using Back-Propagation," Neural and Evolutionary Computation.

[2] Hertz, Krogh & Palmer, Introduction to the Theory of Neural Computation, Chapter ONE: Introduction.

[3] Hertz, Krogh & Palmer, Introduction to the Theory of Neural Computation, Chapter FIVE: Simple Perceptrons (only 5.1, 5.2, 5.4 except Convergence of Gradient Descent, 5.5), Chapter SIX: Multi-Layer Networks (all except 6.5).

[4] Du & Swamy, Neural Networks and Statistical Learning, Chapter 1: Introduction (all except 1.2.2 and 1.4).

[5] Linear regression and Ordinary least squares, Wikipedia. https://en.wikipedia.org/wiki/Linear_regression

[6] Logistic regression, Wikipedia. https://en.wikipedia.org/wiki/Logistic_regression

[7] Naive Bayes classifier, Wikipedia. https://en.wikipedia.org/wiki/Naive_Bayes_classifier

[8] K-Nearest Neighbors, Wikipedia. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[9] S. B. Kotsiantis, D. Kanellopoulos, and P. E. Pintelas, "Data Preprocessing for Supervised Learning," International Journal of Computer Science, vol. 1, no. 2, 2006, ISSN 1306-4428.