**Documentation - Multilayer Perceptron Converter**

The Multilayer Perceptron Converter is a program consisting of five code files that build a decision tree model from a dataset and export the resulting model algorithm in a Structured Text file. Below is a detailed description of each code file:

1. *conversion.py*

This code generates a multilayer perceptron and converts it into a structured text syntax. The following information is requested from the user:

* Dataset file name;
* Name of the .txt file to store the converted algorithm;
* Name for the output variable.

After gathering this information, the script:

* Checks if the .csv file exists, reporting an error otherwise;
* Calls the *preprocess\_and\_generate\_mlp* function from the *mlp\_generator.py* to train and build the Multilayer Perceptron based on the provided dataset;
* Uses the *convert\_network* function from *mlp\_conversion.py* to adjust the decision tree to a Structured Text syntax suitable for application in the PLC.

1. *mlp\_generator.py*

This script is responsible for generating a multilayer perceptron from a dataset file. Using the pandas and scikit-learn libraries, it processes the data and builds the MLP model. The script has a predefined structure that users can modify freely, adjusting the tree parameters as needed. The process mainly includes:

* Building and training a multilayer perceptron model (adaptable for regression or classification);
* Exporting the trained structure to a .txt file, simplifying visualization and interpretation of the model for the user.

1. *mlp\_model.py*

This script inserts the trained model’s structure into a .txt file, simplifying visualization and interpretation of the model for the user. The generated file is used for converting the algorithm to the desired Structured Text syntax.

1. *mlp\_conversion.py*

This script generates a .txt file by converting the output file produced by mlp*\_generator.py*. It contains the main function *convert\_network*, which performs the necessary conversion and additional operations with the support of functions organized in *mlp\_functions.py*. The process includes:

* Translating the algorithm syntax;
* Generating a .txt file containing the MLP algorithm in the Structured Text format;
* Generating a .txt file containing a table with information about the input and output parameters.

1. *mlp\_functions.py*

This script contains essential functions for converting multilayer perceptron into a conditional structure suitable for PLCs. Each function includes a detailed description of its parameters and functionalities:

* *def network\_dictionary(network\_file):*

This function takes the file path as an input parameter, reads it, and creates and returns a dictionary with general information about the network, such as the number of layers and details of each one (neurons, weights, biases, etc.).

* *def write\_file(layers, sizes\_vector, input\_vector, destination\_file, neural\_network, activation\_function, out\_activation\_function):*

This function takes as parameters three vectors: one with layer information, one with the number of neurons in each layer, and another containing input variables. Additionally, it also receives the file path to be read and converted, the dictionary created in \textit{network\_dictionary}, and the activation functions for intermediate layers and the output layers. In addition to modifying and generating the final file, this function returns two vectors with input and output variables that are essential for constructing the table of information necessary to declare the variable in the 800xA program. It also returns a vector with biases to be added in \textit{mlp\_conversion}.