AI Lab 5 part B

CoEvolution

Submitted BY :

Abed Abo Hussien 208517631

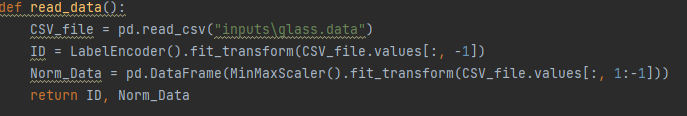
Samer Kharaoba 209050202

Section א :

In inputs folder we can find the glass.data folder

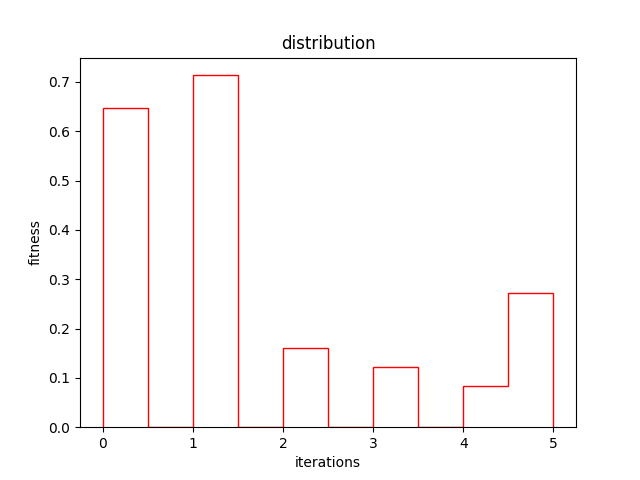
Section ב:

We used a built in function in pandas library :



ID refers to the codes from 0-5

The given Distribution of the data is :

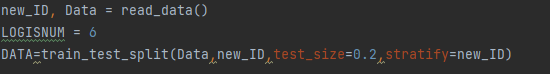


Section ד:

In read\_data() function described above we used built in functions to normalize the data it’s called DataFrame of the Panda library



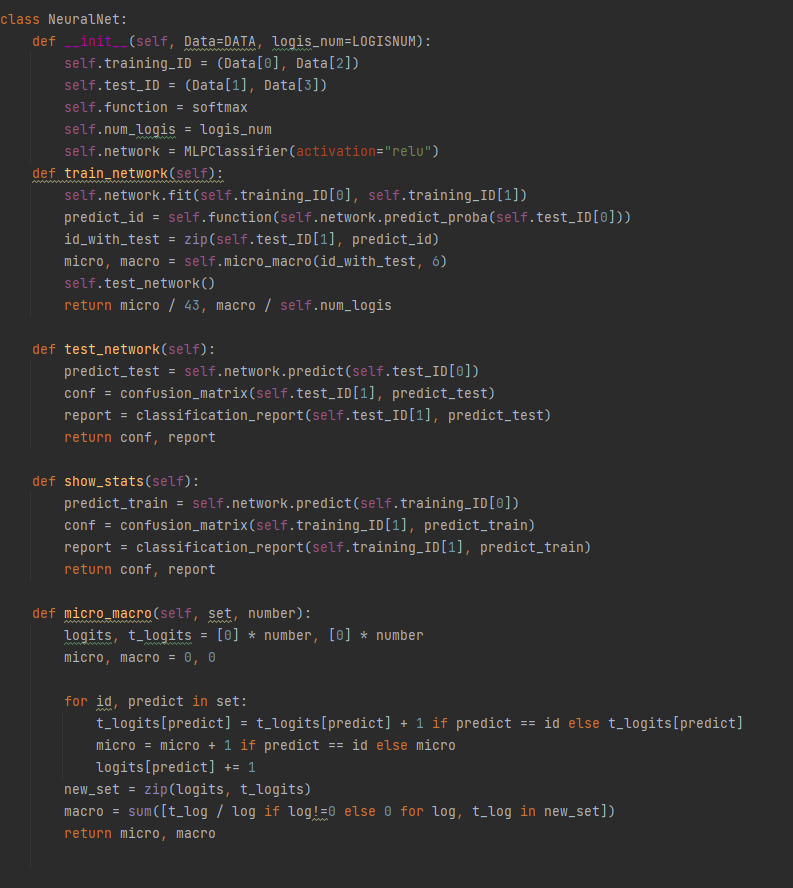
Section ג:



Again here we used a function from the Panda library that splits the data that we got from the CVS file into a training set and a test set , all were put in a variable named DATA

Section ה:

We built a class to create Neural Networks , we used a MLPClassifier that defines the basics of a Neural network and runs according to our wishes:



1. \_\_init\_\_(self,DATA,Logis\_num):

* This class gets Data (devided into 2 groups) and the number of logits that we would like it to have :

1. Test network:

* Uses the test set on the network and returns a report on the outcome

1. Train network:

* Uses the training set to train the network

1. Show stats :

* Returns a report on the training set and it’s confusion matrix

1. Micro macro:

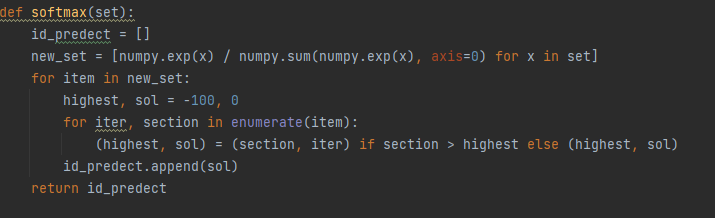
* Calculates macro and micro values of the softmax evaluation function
* The 6 logits that are sent to the softmax function are in the micro macro function

Text

Description automatically generated

Section ו:

Firstly we created a matrix that contains all values of softmax function then worked on all of them to find the solution



Section ז:

We tested on one Neural network described by the class above and the total results were :



By using this simple code :

Text

Description automatically generated

Section ח :

Firstly we created a new agent that has a Neural Network :



* Note: DNA is a variation of our Agent

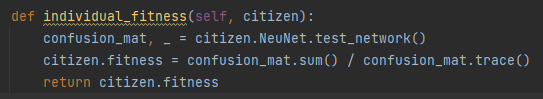
Here we made it so that the neural networks that are created in the population are different from one another :

1. By activation scheme
2. By hidden layer size ,that is decided by random and dictates the depth of the network
3. By the type of solver

Note : logistic returns the sigmoid function

1. Alpha gives the L2 penalty

Fitness function :

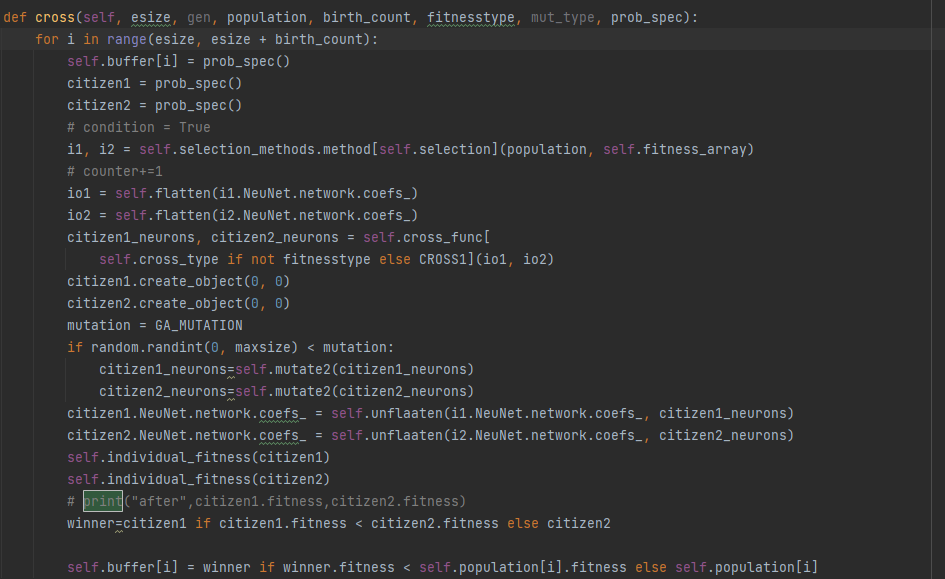


We used the confusion mat to calculate the fitness so basically so that the alpha value that we put in the neural network would have an effect according to the f1 criteria .

As a reminder alpha is the penalty that we used that uses L2 normal on the weights

Sections ט &י :

Firstly we tuned the cross function so that it manipulates the layers of the network:



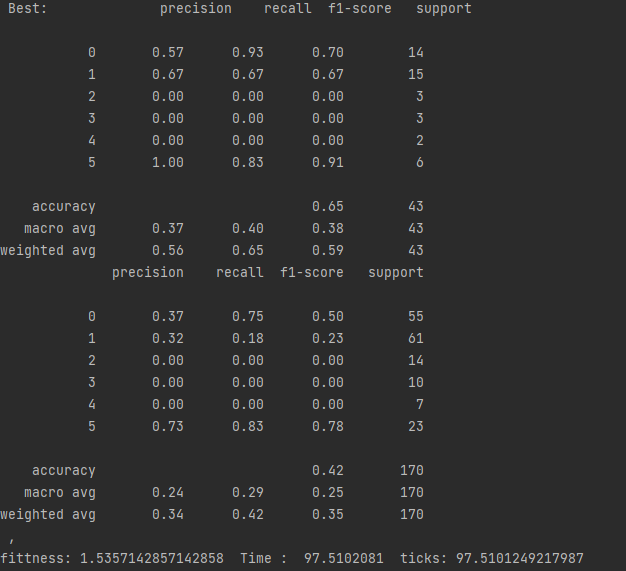


1. we used flatten to flatten the hidden layers ( positions of the weights) that flattens the array of weights into one array



1. Used One point cross function that was Implemented before
2. Used Random Mutate to a random number
3. Unflattened the weights matrix ( Called coefs\_)
4. Calculated fitness for both new weights

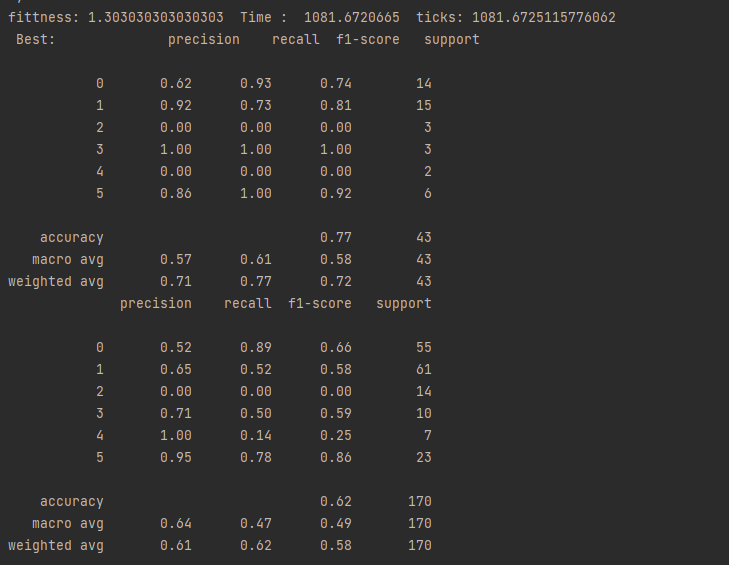
The results :



The above one is the training set

The bottom one is the test set

Best individual :



**Last section :**

**Parallelization :**

The parallelization process would allow us to run more iterations and allows the evolution phase specifically the fitness calculation in those stages to run much faster as long as the memory overhead doesn’t become too much as the parallelization process requires handling threads via a thread pool that can take a respectable amount of memory.

**Caching snippets of similar networks:**

the caching of snippets of a network can help reduce the time it takes to calculate the fitness for similar individuals as such we would have similar sections (fitness values) for those sections on both networks.

**Using integers instead of floating points:**

Using integers does indeed reduce the time of calculations , i.e. faster fitness calculations but using this method we would lose precision thus losing the whole point of Neural Networks .  
so , in these tests we wouldn’t recommend using integers.

**Smart initialization of Weights:**

This one doesn’t help with the overhead from the fitness but would help us converge faster to a solution .thus we sure think that it would be a great to have feature in the algorithm.

Instruction :

Enter the values you are asked to enter

Final thoughts :

* Improve the 3 points alluded to in the last section

References :

<https://scikit-learn.org/stable/modules/neural_networks_supervised.html>

<https://python-course.eu/machine-learning/neural-networks-with-scikit.php>

<https://www.pluralsight.com/guides/machine-learning-neural-networks-scikit-learn>