Assumptions:

1. Function used: Sigmoid.
2. Training data is randomly sampled from the dataset.
3. Error used: Mean Square error.
4. The program knows the values of each attribute.
5. File names should be “cardata.csv”, “irisdata.csv”, “adultdata.csv”. Path can be anything, only condition is that the folders in the path should not contain any spaces. (Screenshot attached in every dataset specific report)

Pre-processing

Inputs:

1. File from where the data is to be read.
2. File from where the data is to be stored.

* This information is stored in Arraylist statically. Whenever the program reads the new instance it will match with the values of the list and store its position as its own value. In this way, categorical data is converted into numerical.
* Numerical data is converted using mean and standard deviation.
* If it encounters “?”, it will ignore that row, and read the next line.
* The final output is kept as it is. That is a regression problem. It is not converted to binary variable.

Training:

Inputs:

* File from where the data is stored i.e. the output file from the preprocessing stage.
* Training Size
* Maximum number of iterations
* Number of hidden layers
* For every hidden layer, number of neurons

1. The program will terminate when either it reaches max iteration or error is less than 1E-6.
2. Value of alpha = 0.5
3. Accuracy is good when the number of layers are more, but not very large.
4. Also, optimal number of iteration = 200

Logic:

* An arraylist is made for every neuron which includes the weights of edges between itself and every neuron of next layer.
* This arraylist of neurons is stored in the new Arraylist of arraylist.
* An arraylist is created for storing X values, Delta values for each neuron. For bias, its X value is 1. Delta value of neurons in input layer and bias is 0.

Testing:

* After training given number of instances and iterating for given iterations, a network will be set which contain all the updated weights.
* In this step, attributes of new instance will be taken as input. Also, we know the actual output from the dataset.
* We will predict the final output with our trained network and given new attributes, and find the output. This will be our predicted output.
* Error is calculated using Mean Square Error.

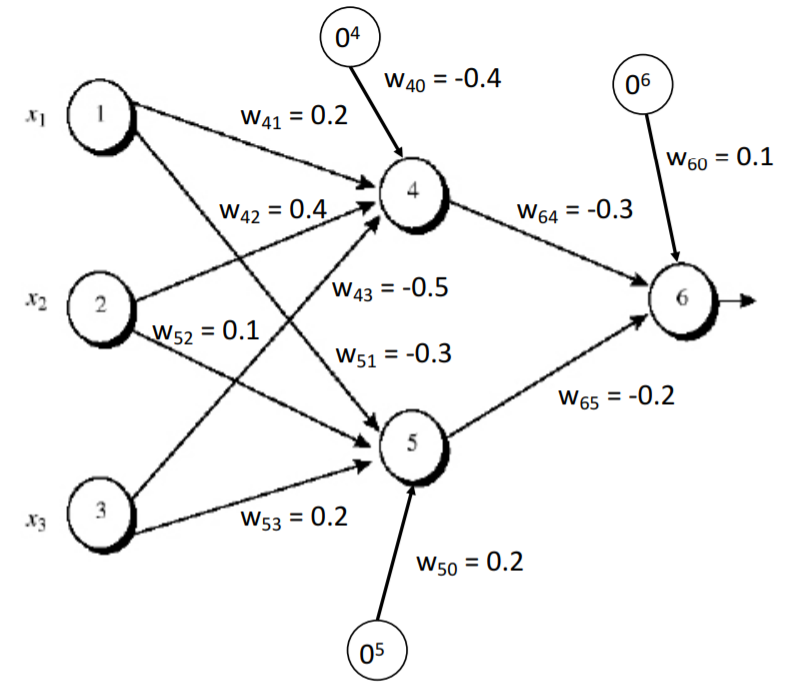
Output:

For every layer (excluding output) it will print

1. The Layer numbers
2. Weights of every neurons in that layer
3. Weight of the bias in that layer.

First layer is input layer.

And later, all are hidden layers starting from 1 to n, where n is the number of hidden layers provided by user.



Consider this network (ignore the weights)

Weights will be randomly initialized.

Output will be shown as

Layer 0

Neuron 1 weights: W41, W51

Neuron 2 weights: W42, W52

Neuron 3 weights: W43, W53

Bias Weight: W40, W50

Layer 1

Neuron 1 weights: W64

Neuron 2 weights: W65

Bias Weight: W60