Effie Bluestone

RUN CODE use the make file:
To compile the code just type make to get both the train.exe and test.exe.
Then you can choose which program to run based on if you call
`./NNtrainAI.exe or ./NNtestAI.exe`

DATA SET TALKS

Created my own test data which consisted of producing 1000 lines of 10 randomly created double numbers using the built-in random function in python. These are representative of the 10 initial weights of the edges to the hidden layers of the network.

The training data was created by looking at each set of 10 numbers grouping first five numbers adding them up and then comparing that to the last 5 and adding them up. If the first 5 are bigger than the last 5 and the first number is also positive the output should be 1 otherwise the output will be 0.
The Neural network that I created analyzed this data and got really good results with only using **200 epochs, a .01 learning rate** and only 3 hidden layers but felt this problem may be too simple. So looked at another data set.
(The previous data set was name with the word **create** in email Files)

This new data set was a bit more difficult. Used a data set from
https://github.com/jbrownlee/Datasets/blob/master/pima-indians-diabetes.csv

```
1.
Title:
Pima
Indians
Diabetes
Database

          2. Sources:
             (a) Original owners: National Institute of Diabetes and Digestive and
                                  Kidney Diseases
             (b) Donor of database: Vincent Sigillito (vgs@aplcen.apl.jhu.edu)
                                    Research Center, RMI Group Leader
                                    Applied Physics Laboratory
                                    The Johns Hopkins University
                                    Johns Hopkins Road
                                    Laurel, MD 20707
                                    (301) 953-6231
             (c) Date received: 9 May 1990
```

Effie Bluestone

3. Past Usage:
    1. Smith,~J.~W., Everhart,~J.~E., Dickson,~W.~C., Knowler,~W.~C., \&
       Johannes,~R.~S. (1988). Using the ADAP learning algorithm to
forecast
       the onset of diabetes mellitus. In {\it Proceedings of the
Symposium
       on Computer Applications and Medical Care} (pp. 261--265). IEEE
       Computer Society Press.

       The diagnostic, binary-valued variable investigated is whether
the
       patient shows signs of diabetes according to World Health
Organization
       criteria (i.e., if the 2 hour post-load plasma glucose was at
least
       200 mg/dl at any survey examination or if found during routine
medical
       care).   The population lives near Phoenix, Arizona, USA.

       Results: Their ADAP algorithm makes a real-valued prediction
between
       0 and 1. This was transformed into a binary decision using a
cutoff of
       0.448. Using 576 training instances, the sensitivity and
specificity
       of their algorithm was 76% on the remaining 192 instances.

4. Relevant Information:
      Several constraints were placed on the selection of these
instances from
      a larger database. In particular, all patients here are females
at
      least 21 years old of Pima Indian heritage. ADAP is an adaptive
learning
      routine that generates and executes digital analogs of perceptron-
like
      devices. It is a unique algorithm; see the paper for details.

5. Number of Instances: 768

6. Number of Attributes: 8 plus class

7. For Each Attribute: (all numeric-valued)

Effie Bluestone

```
   1. Number of times pregnant
   2. Plasma glucose concentration a 2 hours in an oral glucose
tolerance test
   3. Diastolic blood pressure (mm Hg)
   4. Triceps skin fold thickness (mm)
   5. 2-Hour serum insulin (mu U/ml)
   6. Body mass index (weight in kg/(height in m)^2)
   7. Diabetes pedigree function
   8. Age (years)
   9. Class variable (0 or 1)

8. Missing Attribute Values: Yes

9. Class Distribution: (class value 1 is interpreted as "tested positive
for
   diabetes")

   Class Value  Number of instances
   0            500
   1            268

10. Brief statistical analysis:

    Attribute number:    Mean:   Standard Deviation:
    1.                     3.8     3.4
    2.                   120.9    32.0
    3.                    69.1    19.4
    4.                    20.5    16.0
    5.                    79.8   115.2
    6.                    32.0     7.9
    7.                     0.5     0.3
    8.                    33.2    11.8
```

The first thing done with the way to put it all into excel. Then the data for each attribute was normalized by dividing the entire column by the max number of that specific attribute this was make sure all the attributes were less than or equal to 1. The data set was then split into 70% being in the training and the rest in the test set.

Created the initial Neural network file using a zero mean gaussian distribution (which is a normal 0 - 1 distribution) for the initial weights of the edges to the hidden layer and to the outputs of the network. This was made using an online mean gaussian distribution generator (from 0-1).

Effie Bluestone

The initial network was made with first with 4 hidden and varying the epochs and learning rates the layers and was not preforming good at all. So more hidden layers were added. I ended up sticking with 500 epochs with a .1 learning rate with hidden layers of 256 nodes to achieve my results .

This gave the best results out of the things I played around with. The results were still low values so there must be unobservable things at play. Like there are probably other factors which are not included in this set which also cause diabetes. Additionally In this case the attributes have overlap and are not completely independent. For example Body Mass index and diastolic heartrate. As always, more data points could be good for better results.

Overall this was a complex data set which was hard for single layer percepton to learn and do well on.