Supervised Learning (AA 2015)
Classification with Naïve Bayes and MLP

Deadline: April 19, 2015

Introduction

The objective of this assignment is to implement three neural network classifiers, select the best one based on the training data set and then test it against the test data set. You will also train a naïve Bayes classifier on the training set, test it on the test set and compare the results with the selected neural network classifier. Your implementation should allow you to train and use at least three network architectures:

1. A single neuron.
2. A fully connected feedforward network with one hidden layer of 2 neurons and one output neuron.
3. A fully connected feedforward network with one hidden layers of 3 neurons each and one output neuron.

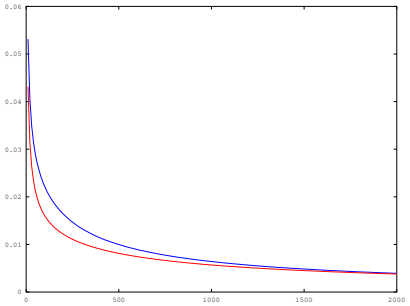
The response function for each neuron should be the sigmoid (logistic) function: $y = \frac{1}{1+e^{-x}}$.

1 Implementation and preliminary tests.

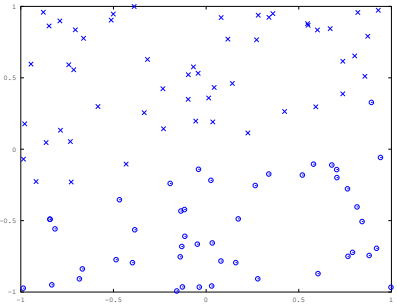
To test your MLP implementation you can use three synthetic data sets generated using the functions supplied: `linearsep`, `xorlike` and `radial`. You can obtain training and test sets for these synthetic data running the `generate` script provided.The first one is linearly separable, the other two are not. With a single meuron it is possible to classify the `linearsep` data set with low error. The figure below shows the training and test error using 100 points and the classification of each point for the training and test sets. Crosses are class 0, circles are class 1 and red symbols indicate a classification error. The error values on the first panel are the mean squared errors of the network prediction:

$$E = \frac{1}{N}\sum_{n=1}^{N}(y(x_n) - t_n)^2$$

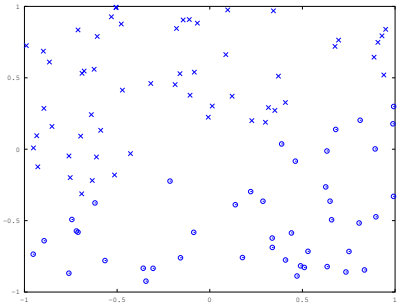
where x_n is one point, N the total number of points, $y(x_n)$ the network response and t_n the true class of the point. The red line shows the training error, the blue line the test error. The x axis indicates the number of epochs of training (one epoch is one passage through all the training set).



Error during training

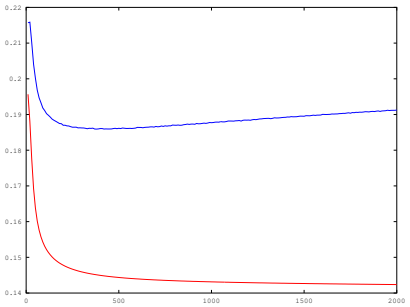


Classification (training set)

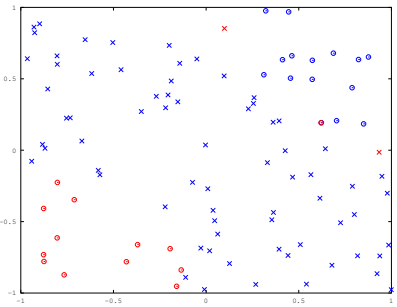


Classification (test set)

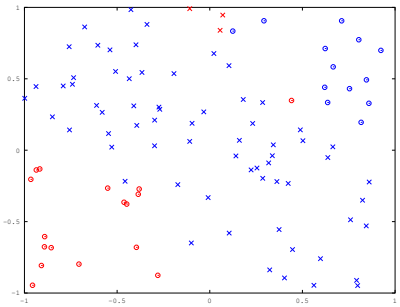
A single neuron, however, cannot adequatly classify the `xorlike` data sets or the `radial` data set:



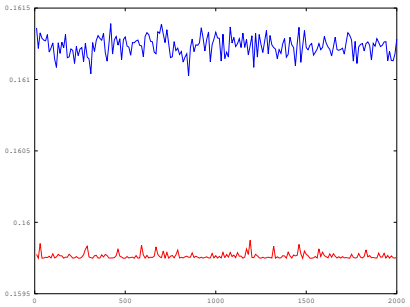
Error during training



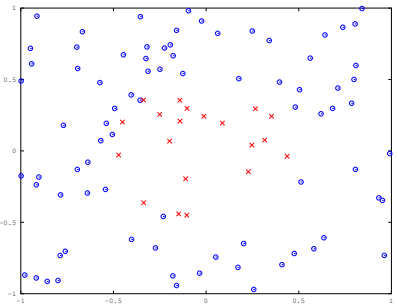
Classification (training set)



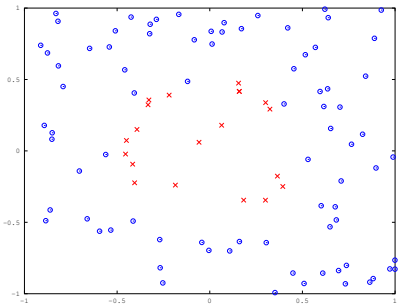
Classification (test set)



Error during training

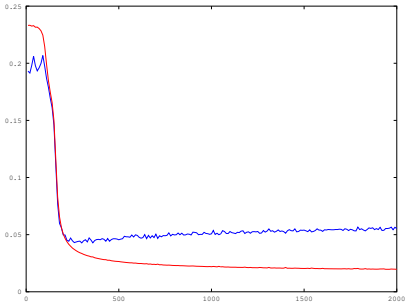


Classification (training set)

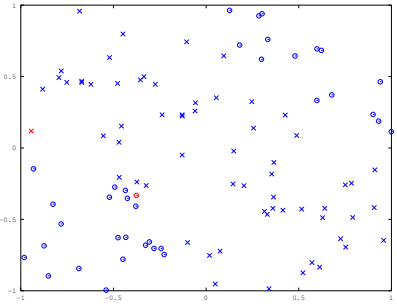


Classification (test set)

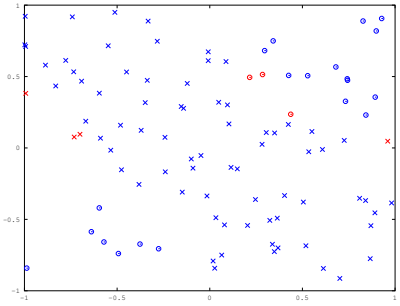
With one hidden layer of 2 neurons it is possible to classify the `xorlike` data set with little error:



Error during training

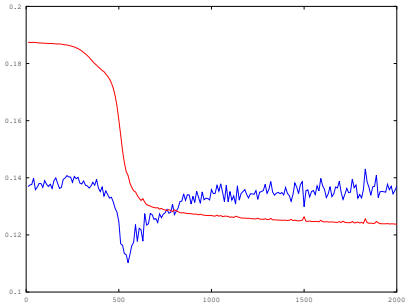


Classification (training set)

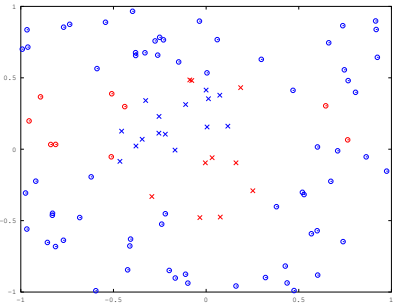


Classification (test set)

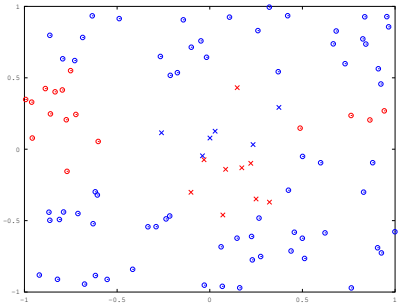
However, this is still not enough for a good classification of the `radial` data set:



Error during training

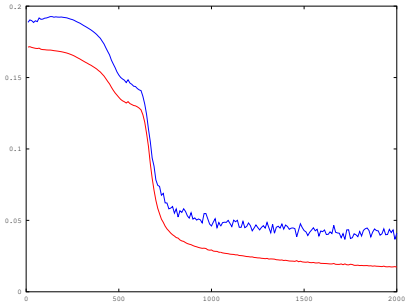


Classification (training set)

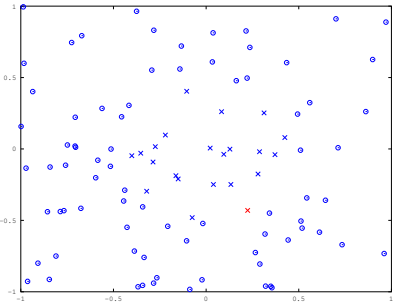


Classification (test set)

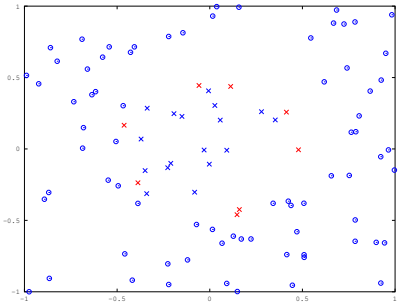
To properly classify the third data set it is necessary to increase the number of hidden neurons to 3:



Error during training



Classification (training set)



Classification (test set)

For this first part you must test your implementation using these artificial data sets. Present the graphs indicating your training and test errors and also the confusion matrices computed on the training and test sets for each of the three models and artificial data sets.

2 Application and model selection

The data set for the application was previously split into a training set (`bcdata_train.txt`) and a test set (`bcdata_test.txt`). These data were derived from the Wisconsin Breast Cancer Database, available at the UCI machine learning repository. These files contain one data point per line, corresponding to one breast tissue sample. All attributes are integers. The first attribute is an identification number and should be discarded. The last attribute is the class of the tissue sample: 2 for benign, 4 for malignant. For more details on the attributes consult the file `breast-cancer-wisconsin.names`.

2.1 Preprocessing

Neural networks are easier to train when input values are close to zero. You should rescale the input vectors so the features span values from 0 to 1 or from -1 to 1. Furthermore, the class values 2 and 4 in the data set files are not useful values for comparing with the network response, which ranges from 0 to 1 with the logistic function. So these values should also be transformed to classes 0 and 1.

2.2 Model selection: neural network classifiers

Using the training set, you must select the best neural network model for this problem. You should estimate the generalization error, take into account the complexity of the model and also try to

minimize the generalization error by choosing a suitable number of epochs during training. For this task consider the 3 models implemented in the first part. Remember that model selection should be done with cross-validation.

2.3 Comparing with the Naïve Bayes Classifier.

Train both your best neural network model and a naïve Bayes classifier using the full training set and use both to classify the test set. For each model build a confusion matrix specifying the false negatives and false positives, calculate the error rate and estimate the 95% confidence interval for the number of errors in a data set of the same size as the test set. Use the confidence intervals to decide if the performance difference between the neural network and the naïve Bayes classifier is statistically significant. Check this estimate with McNemar's test.

3 Report format.

You should submit your work in a .zip file containing a folder with your source code and your report in a pdf file. The report should be approximately 15 pages long, containing the following sections:

1. Introduction, explaining the two different classifiers (Naïve Bayes and feedforward neural networks).
2. Implementation, explaining the important parts of the implementation like backpropagation, training of the Bayes classifier and processing the input values and the training set, and also demonstrating the correctness of your implementation on the synthetic data.
3. Method, explaining the training and model selection phase.
4. Results and Discussion showing the results obtained, justifying the selection of the neural network and the comparative analysis of the two classifiers.

The source code folder must contain the following three scripts:

1. `implementation` to test your implementation, showing the results of the 3 neural networks for the 3 synthetic data sets.
2. `selection` to run the model selection phase, showing the results for the 3 neural networks and the appropriate charts.
3. `comparison` to train the two final models (ANN and naïve Bayes), test them against the test set and show their evaluation and comparison.