CS 383 - Machine Learning

Assignment 8 - Artificial Neural Networks
Winter 2017

Introduction

In this assignment we'll implement non-linear classifier, an Artificial Neural Network, and look at using it for binary classification, varying precision and recall, and direct multi-class classification.

Grading

Although all assignments will be weighed equally in computing your homework grade, below is the grading rubric we will use for this assignment:

Part 1 (Binary ANN)	30pts
Part 2 (Precision-Recall)	15pts
Part 3 (Multi-class ANN)	15pts
Report	5pts
TOTAL	65pts

Datasets

Spambase Dataset (spambase.data) This dataset consists of 4601 instances of data, each with 57 features and a class label designating if the sample is spam or not. The features are *real valued* and are described in much detail here:

https://archive.ics.uci.edu/ml/machine-learning-databases/spambase/spambase.names

Data obtained from: https://archive.ics.uci.edu/ml/datasets/Spambase

Cardiotocography Dataset (CTG.csv)

Download the file CTG.csv from Bblearn. This file contains 2126 instances of 21 feature pertaining to information obtained from Cardiotocography tests. Our task is to determine the fetal state class code given an observation. This code can be one of the 3 values and pertains to the LAST column of the dataset. The second to last column of the dataset can also be used for classification but for our purposes DISCARD it.

Your scripts that use this dataset must be able to run on any dataset where the first two rows contain header information, the 2nd to last column is to be discarded, and the last column contains the target class.
You can read more about the dataset here:

http://archive.ics.uci.edu/ml/datasets/Cardiotocography

1 Artificial Neural Networks

First download the dataset *spambase.data* from Blackboard. As mentioned in the Datasets area, this dataset contains 4601 rows of data, each with 57 continuous valued features followed by a binary class label (0=not-spam, 1=spam). There is no header information in this file and the data is comma separated. As always, your code should work on any dataset that lacks header information and has several comma-separated continuous-valued features followed by a class id $\in 0, 1$.

You are to create, train, and test an artificial neural network containing a single hidden layer. You are to do this *from scratch*, not using any ANN library functions.

Write a script that:

1. Reads in the data.
2. Randomizes the data.
3. Selects the first 2/3 (round up) of the data for training and the remaining for testing
4. Standardizes the data (except for the last column of course as well as the bias feature) using the training data
5. Trains an artificial neural network using the training data
6. During the training process, compute the *training error* after each iteration. You will use this to plot the training error vs. iteration number.
7. Classifies the testing data using the trained neural network.
8. Computes the testing error.

Implementation Details

1. Seed the random number generate with zero prior to randomizing the data
2. Make sure to add a bias input node.
3. Set the learning parameter $\alpha = 0.5$.
4. There should only be a single hidden layer.
5. The hidden layer size should be 20, although this should be a variable parameter.
6. Do *batch* gradient descent.
7. Initialize all weights to random values in the range $[-1, 1]$.
8. Do 1000 training iterations.
9. Since this is binary classification, you should only have one output node.
10. Consider a sample to be positive (Spam) if the output node has a value > 0.50 and negative (Not Spam) otherwise.
11. Compute the testing error as $TestError = 1 - \frac{\#testsamplescorrect}{totalnumberoftestsamples}$.

In your report you will need:

1. The test error
2. A plot of *training* error vs iteration.

Testing Error	0.0841

Table 1: ANN Testing Error

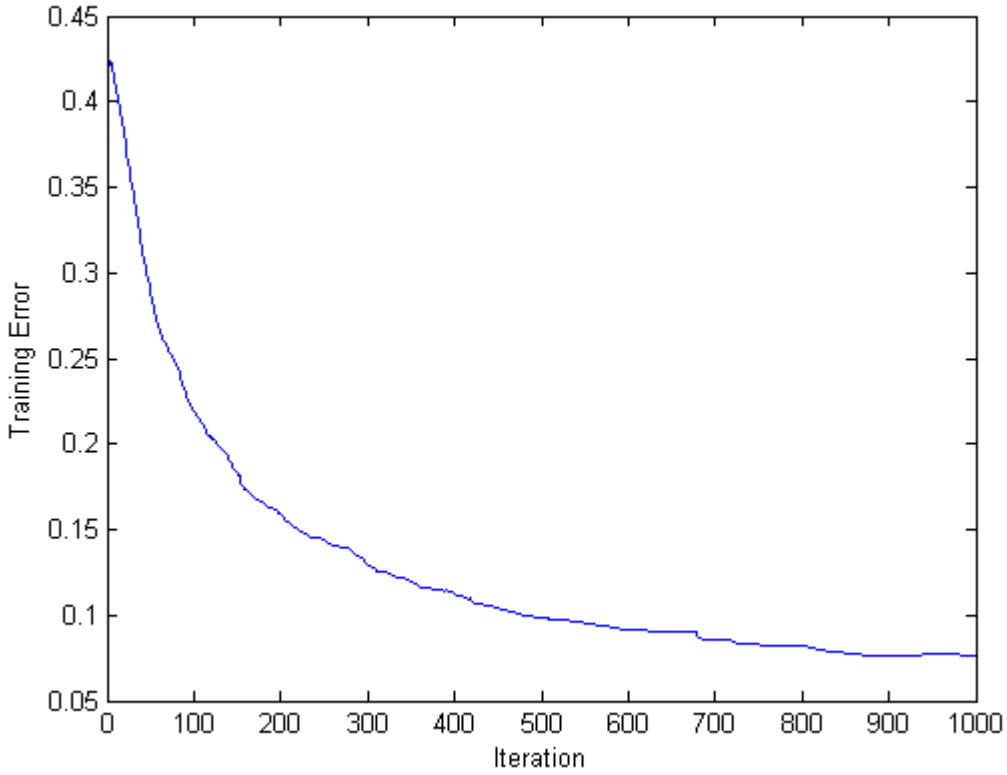

Figure 1: Training Error for ANN

2 The Precision-Recall Tradeoff

Next we want to look at the precision-recall tradeoff by varying the threshold required to make an observation be considered spam.

In the previous part your ANN classifier should have output a *likelihood* of being Spam and if that likelihood is greater that 50% we said that the observation is predicted to be spam.

Now let's vary that threshold t (which was previously 0.5) from 0.0 to 1.0 in increments of 0.1, each time computing the precision and recall by labeling observations as *Positive* if their likelihood is greater that the current t and *Negative* otherwise.

Finally we'll graph these $(precision, recall)$ pairs to form a Precision-Recall graph.

Implementation Details The same as in the prior problem.

In your report you will need:

1. Your plot of precision vs recall on the testing set.

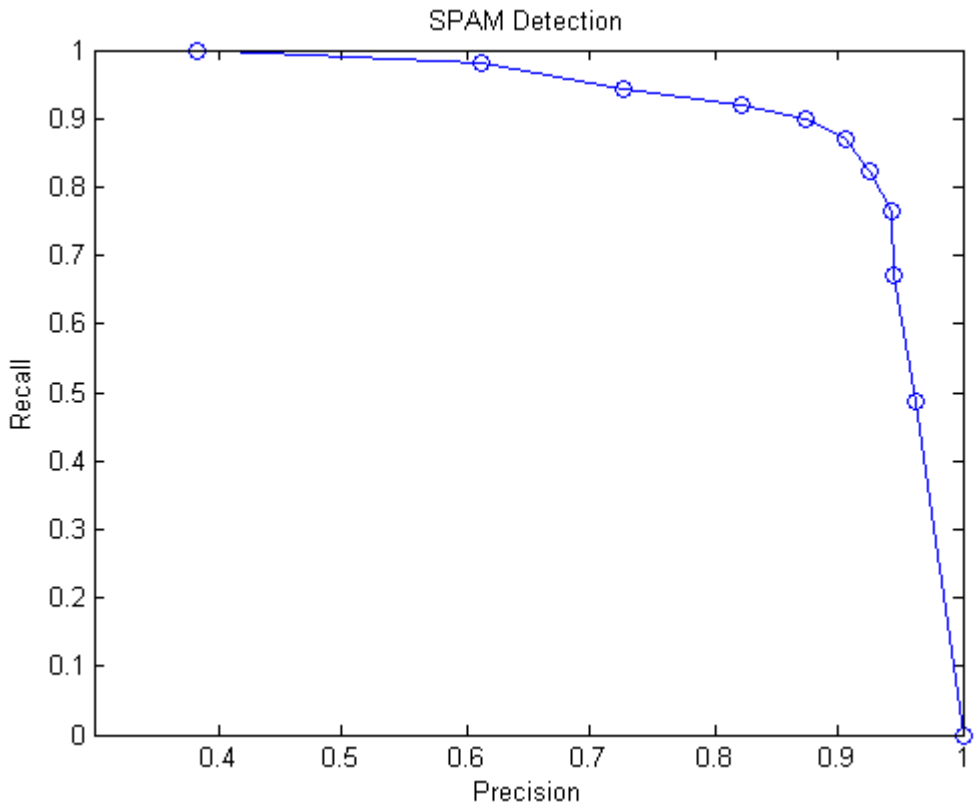

Figure 2: PR-Graph for ANN

3 Multi-Class Artificial Neural Networks

Download the *CTG.csv* data set from Blackboard. Your task is to determine the fetal state class code given an observation. This code can be one of three values (multi-class!) and pertains to the *last* column of the dataset. The second to last column of the dataset can also be used for classification, but for our purposes *discard it*.

You are to create, train, and test an artificial neural network containing a single hidden layer. You are to do this *from scratch*, not using any ANN library functions.

Implementation Details Same as the other parts except...

1. Since there are three possible classes, we'll do multi-class classification directly by having an output layer have three nodes. But this also should be a variable size based on the number of potential output values. To choose the class label find the output node with the largest value.

In your report you will need:

1. The test error
2. A plot of *training* error vs iteration.

Testing Error	0.1142

Table 2: ANN Testing Error

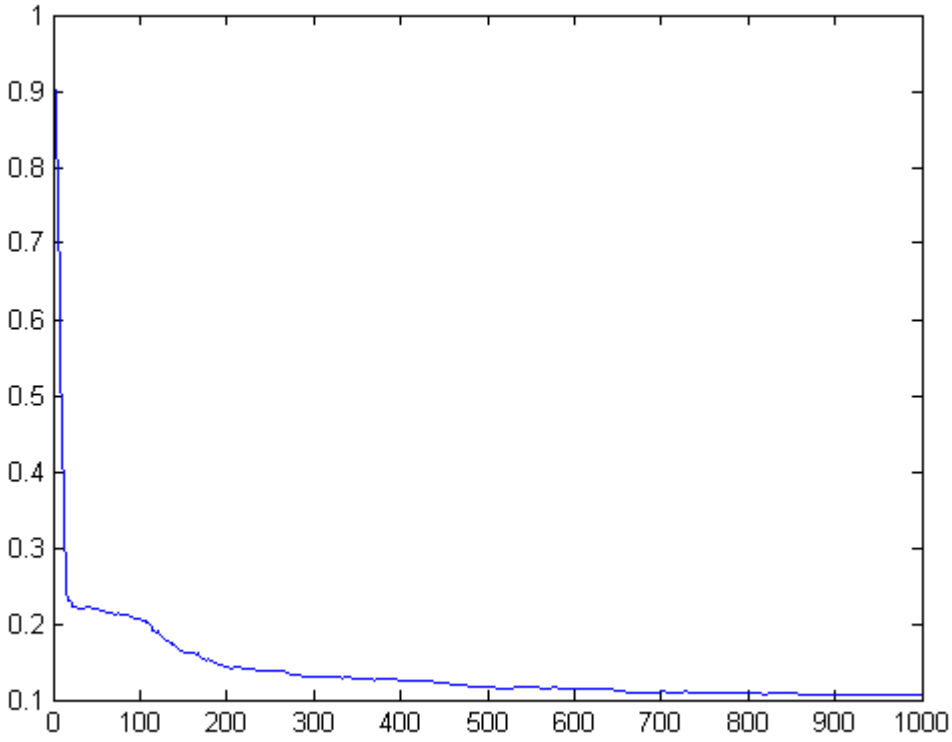

Figure 3: Training Error for Multi-Class ANN

Submission

For your submission, upload to Blackboard a single zip file containing:

1. PDF Writeup
2. Source Code
3. readme.txt file

The readme.txt file should contain information on how to run your code to reproduce results for each part of the assignment.

The PDF document should contain the following:

1. Part 1:
   (a) Requested Statistical Information
   (b) Plot of training error vs iteration.
2. Part 2:
   (a) Precision-Recall plot.
3. Part 3:
   (a) Requested Statistical Information
   (b) Plot of training error vs iteration.