Homework Assignment 1

CSE 253: Neural Networks

Winter 2016

Instructions

Due Tuesday, January 12^{th}:

1. Please hand in a hard copy for your assignment. We prefer a report written using LaTeXin NIPS format for each assignment. You are free to choose an alternate method (Word, etc.) if you want, but we still prefer NIPS format.

2. You may use a language of your choice (Python and MATLAB are recommended) but all source code used in the assignment must be attached in the appendix. Please keep the code clean with explanatory comments, as they may be reused in the future.

3. Using the MATLAB toolbox for neural networks or any off-the-shelf code is strictly prohibited.

4. Please do your own work. You can discuss the assignment with other classmates, as long as you follow the Gilligan's Island Rule (see below). Books, notes, and Internet resources can be consulted, but not copied from. Working together on homeworks must follow the Gilligan's Island Rule (Dymond, 1986): No notes can be made during a discussion, and you must watch one hour of Gilligan's Island or equivalently insipid activity before writing anything down. Suspected cheating will be reported to the Dean.

Problems from Bishop (40 points)

Work problems 1-4 (10 points each) on pages 28-30 of Bishop.

Perceptron (25 points)

1. (*5 points*) Recall the perceptron activation rule:

$$y = \begin{cases} 1 & \text{if } \sum_{i=1}^{d} w_i x_i \geqslant \theta \\ 0 & \text{else} \end{cases}$$

We can consider this as a linear discriminant where the input vector x categorized as C_1 if (abusing our notation slightly) $y(x) = w^T x - \theta \geq 0$ and class C_2 otherwise. Assuming $d = 2$, derive the equation for the decision boundary (a line), i.e., the equation for $y(x) = 0$ (*2 points*), and prove that the distance from the decision boundary to the origin is given by (*3 points*):

$$l = \frac{-w_0}{\|w\|}$$

2. (*7 points*) In class, we learned the "OR" function using the perceptron learning rule. Now suppose we want to learn "NAND" using the four patterns shown in Table 1. Please complete the following task:

Input	Output
0 0	1
0 1	1
1 0	1
1 1	0

Table 1: The "NAND" function

(a) Write down the perceptron learning rule. (*1 point*)

(b) Create a table with 7 columns: | x_1 | x_2 | Output | Teacher | w_1 | w_2 | Threshold (θ) |. Initialize w_1, w_2 and θ to 0 and assume the learning rate is 1. Add one row for each randomly selected pattern (training example) for the perceptron to learn. Stop when the learning converges. Please present the resulting table and show the final learned weights and threshold. You may pick the order to make the learning converge faster, if you can.(*4 points*)

(c) Is the solution unique? Why or why not? (*2 points*)

3. **Programming Assignment (*13 points*). The data is in hw1_iris.tar.gz under resources on piazza** In this problem, we will attempt to make a flower classifier. We shall use a modified version of the popular UCI Iris dataset provided in the file *iris_train.data* within the iris folder. The file describes a data set with the following characteristics:

(a) Number of Attributes: 4 numeric attributes and the class label.

(b) Attribute Information:

i. sepal length in cm
ii. sepal width in cm
iii. petal length in cm
iv. petal width in cm

(c) The last column is the class label (Iris-setosa, Iris-versicolor)

(d) Your tasks are:

i. Z-score the each attribute (feature) of data by the equation:

$$\tilde{x}_i^n = (x_i^n - \mu_{x_i})/\sigma_{x_i}$$

where μ_{x_i} and σ_{x_i} are the mean and standard deviation of the i^{th} input variable, respectively, over the training set. Z-scoring is a common pre-processing step when dealing with datasets. Can you think of why it is useful in our application? (*2 points*)

ii. Using the MATLAB function scatter(X,Y), plot each of the six 2 dimensional feature spaces (i.e., sepal length vs. sepal width, sepal width vs. petal width, etc.) using a different symbol for each class. Put these graphs in your report. Are the classes linearly separable in each of the feature spaces? Why? (*3 points*)

iii. Train a perceptron to classify the data using the delta rule on the training set *iris_train.data*, using all four features. You are free to choose the stopping criterion when necessary (i.e., a total number of training steps or a test that the classification error (number of incorrectly labeled patterns) is no longer decreasing). Note that if the data are not linearly separable, the perceptron algorithm will never converge, so some stopping criterion will be necessary in that case. Be sure to include your source code in the appendix. (*4 points*)

iv. Classify the test data in *iris_test.data* and report the error rate. The error rate is simply the percentage of misclassified data points in the test set. *NOTE:* You will need to z-score the test data using the corresponding means and standard deviations from the training set. (*2 points*)

v. What learning rate did you use? What happens if you raise/lower the learning rate? (e.g., 2, 1, .5, .25, etc.) Explain your results. (*2 points*)

Logistic and Softmax Regression (25 points)

In this problem, we will classify handwritten digits from Yann LeCun's at MNIST Database. Please download the four files found there, these will be used for this problem. To reduce computation time, we will subset these files. Please use only the first 20,000 training images/labels and only the first 2,000 testing images/labels. *Hint:* Much of the derivations are already done for you in Bishop Chapter 6. You have to fill in the missing steps.

Logistic Regression

Although we consider logistic regression to be a classification technique, it is called "regression" because it is used to fit a continuous variable: the probability of the category, given the data. Logistic regression can be modeled as using a single neuron reading in an input vector $x \in \mathbb{R}^d$ and parameterized by weight vector $w \in \mathbb{R}^d$, where the neuron outputs the probability of the class of x being C_1.

$$P(x \in C_1|x) = g_w(x) = \frac{1}{1 + \exp(-w^\top x)} \tag{1}$$

$$P(x \in C_2|x) = 1 - P(x \in C_1|x) = 1 - g_w(x), \tag{2}$$

where $g_w(x)$ simply notes that the function g is parameterized by w. Note we identify the output y^n of the "network" for a particular example, x^n, with $g_w(x^n)$, i.e., $y^n = g_w(x^n)$. With the hypothesis function defined, we now use the cross entropy loss function (Equation 3) for two categories over our training examples. This equation measures how well our hypothesis function g does over the N data points,

$$E(w) = -\sum_{n=1}^{N} \{t^n \ln y^n + (1 - t^n) \ln(1 - y^n)\}. \tag{3}$$

Here, t^n is the *target* or *teaching signal* for example n. Finally, we may seek to optimize this cost function via gradient descent.

Softmax Regression

Softmax regression is the generalization of logistic regression for multiple (c) classes. Now given an input x^n, softmax regression will output a vector y^n, where each element, y_k^n represents the probability that x^n is in class k.

$$y_k^n = \frac{exp(a_k^n)}{\sum_{k'} exp(a_{k'}^n)} \tag{4}$$

$$a_k^n = w_k^T x^n \tag{5}$$

Here, a_k^n is called the *net input* to output unit y_k. Note each output has its own weight vector w_k. With our model defined, we now define the *cross-entropy* cost function for multiple categories in Equation 6

$$E = -\sum_n \sum_{k=1}^{c} t_k^n \ln y_k^n \tag{6}$$

Gradient Descent

Recall the gradient descent iterative algorithm.

Algorithm 1 Gradient Descent

```
procedure GRADIENT DESCENT
    w_0 ← 0
    for t = 0, ..., m do
        w_{t+1} = w_t − η Σ_{n=1}^{N} ∇E^n(w)
    return w
```

where η is the step size and $\nabla E^n(w)$ is the gradient of the cost function with respect to the weights w on the n^{th} example.

Problem (25 points)

1. **Derive the gradient for Logistic Regression.** *(2 points)* We need the gradient of the cost function, Equation 3, with respect to the parameter θ. Show that for the logistic regression cost function, the gradient is:

$$-\frac{\partial E^n(w)}{\partial w_j} = (t^n - y^n)x_j^n \tag{7}$$

   Show work.

2. **Derive the gradient for Softmax Regression.** *(3 points)* For softmax regression cost function, Equation 6, show that the gradient is:

$$-\frac{\partial E^n(w)}{\partial w_{jk}} = (t_k^n - y_k^n)x_j^n \tag{8}$$

   Show work. **Note:** Here we are departing from Bishop's notation. w_{jk} is the weight from unit j to unit k, not vice-versa.

   Hint: Recall your logarithm rules, such as $\ln(\frac{a}{b}) = \ln a - \ln b$. The hardest part here is the derivative of the softmax.

3. **Read in Data**. Read in the data from files. Each image is 28×28 pixels, so now our unraveled $x \in \mathbb{R}^{784}$, where each vector component represents the greyscale intensity of that pixel. For each image, append a '1' to the beginning of each x-vector; this represents the input to the bias, which we represent as w_0.

4. **Logistic Regression via Gradient Descent.** *(10 points)* Now, using the gradient derived for Logistic Regression cross entropy loss, use gradient descent to classify given $x \in \mathcal{R}^{785}$. We will use a one-vs.-all strategy, where there are 10 different regressions, one for each digit, and the target is 1 if it is that digit, and 0 otherwise. So for instance, if you are classifying $\{2\}$, you would designate an input x that is a "2" as the positive class, and the other nine digits as the negative class. For each image, do this 2-way classification for all ten digits.

   (a) Report the test accuracy for each of the 10 2-way classifications on the test set.

   (b) For each image in the test set, report the overall test accuracy. An example will be considered labeled correctly if the logistic regression classification of the true label has the highest probability. So for instance, if the true label was $\{2\}$ for an image, you would count it as correctly classified if the logistic regression test of $\{2\}$ vs. $\{0, 1, 3, 4, 5, 6, 7, 8, 9\}$ had the highest probability of all the 10 2-way classifications.

5. **Softmax Regression via Gradient Descent.** *(10 points)* Now, using the gradient derived for softmax regression loss, use gradient descent to perform 10-way classification.

   (a) Plot the training accuracy on the training set vs. number of iterations of gradient descent.

   (b) Report the test accuracy on the test set.

   (c) Is the test accuracy lower or higher than the one-vs-all logistic regression approach?