CS 188 Fall 2014 — Introduction to Artificial Intelligence — Written HW10

INSTRUCTIONS

- **Due:** Monday, November 24th, 2014 11:59 PM
- **Policy:** Can be solved in groups (acknowledge collaborators) but must be written up individually. However, we strongly encourage you to first work alone for about 30 minutes total in order to simulate an exam environment. Late homework will not be accepted.
- **Format:** Submit the answer sheet pdf containing your answers. Page 1 must be this page, with your name, SID, and gradescope email filled in. You should solve the questions on this handout (either through a pdf annotator, or by printing, then scanning; we recommend the latter to match exam setting). Alternatively, you can typeset a pdf on your own that has answers appearing in the same space (check edx/piazza for latex templating files and instructions). **Make sure that your answers (typed or handwritten) are within the dedicated regions for each question/part. If you do not follow this format, we may deduct points.**
- **How to submit:** Go to www.gradescope.com. Log in and click on the class CS188 Fall 2014. Click on the submission titled Written HW 10 and upload your pdf containing your answers. If this is your first time using gradescope, you will have to set your password before logging in the first time. To do so, click on "Forgot your password" on the login page, and enter your email address on file with the registrar's office (usually your @berkeley.edu email address). You will then receive an email with a link to reset your password.

Last Name	Lee
First Name	Charles
SID	23728976
Email	charleslee94@berkeley.edu
Collaborators	Daniel He

For staff use only

Q. 1	Q. 2	Total
/20	/30	/50

1. (20 points) Decision Trees

You are given the following data sets

(i)

X_1	X_2	Y
1	1	+
4	5	+
4	5	-
5	5	+

(ii)

X_1	X_2	Y
1	1	+
4	3	+
4	5	-
5	5	+

(iii)

X_1	X_2	Y
1	1	+
4	2	-
4	5	-
5	5	+

(a) [3 pts] Which data sets are linearly separable?

(i) and (ii)

(b) [3 pts] Which data sets have label noise?

(i)

(c) [3 pts] Which data sets can be fit exactly by a decision tree?

(ii) and (iii)

(d) [5 pts] A 1-decision-list is a decision tree in which the "yes" branch of every binary test is a leaf node. For a continuous attribute X_j, a test can be either $X_j > c$ or $X_j < c$. Continuous attributes can appear in multiple tests. Pick a data set and show a decision list that fits it exactly.

(ii): if $X_2 < 4$ then + else if $X_1 > 4$ then + else -

(e) [6 pts] In the absense of label noise, can any two-class data set in two dimensions be fit exactly by a decision list? Briefly explain why, or give a counterexample.

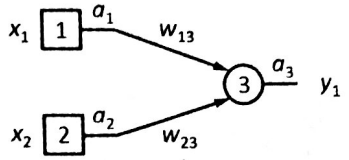
No. They will have problems with an xor problem.

2. (30 points) Neural Networks

In this problem, you are given a simple network that uses the simple linear function $g(x) = mx + b$ (where m and b values are fixed) as the activation function (rather than, for example, a sigmoid function). You will need to write the L_2 norm (squared) loss function, the partial derivative of the loss function, and the gradient descent update rule for certain weights.

Single Neuron Example

We have given you the answers for the loss function, partial derivative, and weight update rule for the following single neuron example. This example should help you understand how to structure your answers to the questions about the slightly more complex network on the following page.



$a_1 = x_1$

$a_2 = x_2$

$a_3 = m(w_{13}a_1 + w_{23}a_2) + b$

Loss function

$$\begin{aligned} Loss(\boldsymbol{w}) &= ||\boldsymbol{y} - \boldsymbol{h_w}(\boldsymbol{x})||_2^2 \\ &= (y_1 - a_3)^2 \\ &= (y_1 - m(w_{13}a_1 + w_{23}a_2) - b)^2 \\ &= (y_1 - m(w_{13}x_1 + w_{23}x_2) - b)^2 \end{aligned}$$

Loss partial derivative

$$\begin{aligned} \frac{\partial}{\partial w_{13}} Loss(\boldsymbol{w}) &= -2(y_1 - a_3)\frac{\partial a_3}{\partial w_{13}} \\ &= -2(y_1 - a_3)ma_1 \\ &= -2(y_1 - m(w_{13}x_1 + w_{23}x_2) - b)mx_1 \end{aligned}$$

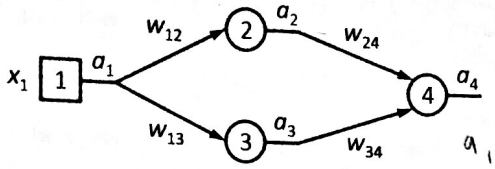
Weight update rule

$$\begin{aligned} w_{13} &\leftarrow w_{13} - \alpha\frac{\partial}{\partial w_{13}} Loss(\boldsymbol{w}) \\ &= w_{13} + \alpha 2(y_1 - m(w_{13}x_1 + w_{23}x_2) - b)mx_1 \end{aligned}$$

(Question continued on next page)

Simple Neural Network with Linear Activation Function

Given the following neural network (again, using the simple linear activation function $g(x) = mx + b$):



$a_2 = m(x_1 w_{12})$

$a_1 = x_1$ $a_3 = m(x_1 w_{13})$

(a) [5 pts] Write the loss function, $Loss(\boldsymbol{w})$, in terms of x_1, y_1, w_{12}, w_{13}, w_{24}, w_{34}, m, and b.

$Loss(w) = (y_1 - a_4)^2$

$(y_1 - m(a_2 w_{24} + a_3 w_{34}) - b)^2$

$(y_1 - m(w_{24} m(w_{12} x_1) + w_{34} m(x_1 w_{13})) - b)^2$

(b) [5 pts] Write the derivative of the loss function with respect to w_{24}, $\frac{\partial}{\partial w_{24}} Loss(\boldsymbol{w})$, in terms of x_1, y_1, w_{12}, w_{13}, w_{24}, w_{34}, m, and b.

$\frac{\partial}{\partial w_{24}} Loss(w) = -2(y_1 - a_4) \frac{\partial a_4}{\partial w_{24}} = 2(y_1 - a_4) m a_2$

$= -2(y_1 -$ $= \frac{\partial}{\partial w_{24}} Loss(w) = -m^2 a_2$

(c) [2 pts] Write the gradient descent update rule for w_{24} with step size α in terms of α x_1, y_1, w_{12}, w_{13}, w_{24}, w_{34}, m, and b.

$w_{24} \leftarrow w_{24} - \alpha \frac{\partial}{\partial w_{24}} Loss(w) = w_{24} + \alpha m^2 w_{12} x_1$

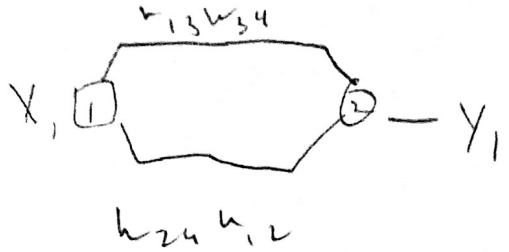
(d) [5 pts] Write the derivative of the loss function with respect to w_{12}, $\frac{\partial}{\partial w_{12}} Loss(\boldsymbol{w})$, in terms of x_1, y_1, w_{12}, w_{13}, w_{24}, w_{34}, m, and b.

$-m^2 w_{24} x$

(e) [2 pts] Write the gradient descent update rule for w_{12} with step size α in terms of α x_1, y_1, w_{12}, w_{13}, w_{24}, w_{34}, m, and b.

$w_{12} + \alpha m^2 w_{24} x$

(f) [3 pts] Because this network has a linear activation function, there is an equivalent network that has no hidden layers. 1) Draw a new (very simple) network that has no hidden layers but computes exactly the same function. 2) Write the new weight explicitly in terms of the w_{12}, w_{13}, w_{24}, w_{34}, m, and b. 3) You will need to adjust the linear activation function, $g_2(x) = m_2x + b_2$. Write the new m_2 and b_2 values in terms of w_{12}, w_{13}, w_{24}, w_{34}, m, and b.



$m_2 = m^2$

$b_2 = 2mb - b$

General Neural Network with Linear Activation Function

Consider a new neural network with n input nodes, n output nodes, one hidden layer with h nodes, and the linear activation function $g(x) = mx + b$ at each hidden and output node. The nodes between each layer are fully connected with weights w_{ij} from the i-th input node to the j-th hidden node and weights w_{jk} from the j-th hidden node to the k-th output node.

(g) [5 pts] Because this general network has a linear activation function, there is an equivalent network that has no hidden layers that computes exactly the same function. 1) Write an equation for the weight w_{ik} from the i-th input node to the k-th output node explicitly in terms of the previous network weights (w_{ij}, w_{jk}), m, and b. 2) You will need to adjust the linear activation function, $g_2(x) = m_2x + b_2$. Write the new m_2 and b_2 values in terms of the previous network weights (w_{ij}, w_{jk}), m, and b.

a $w_{ik} = w_{ij}w_{jk}$

b $m_2 = m^{i+1}$

$b = ((i+1)m - 1)b$

(h) [3 pts] What effect does removing the hidden layer from this general network have on the number of weights? Specifically, in terms of n and h, how many weights are there before and after removing the hidden layer? Discuss in particular the case when $h << n$.

reduces the # of weights

$\frac{n}{h}$