Homework V-KEY

I. Consider the training data given below, x is the attribute and y is the class variable.

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
y	A	A	A	A	B	A	A	A	A	B	B	B	B	A	B	B	B	B

I. What would be the classification of a test sample with x = 4.2 according to 1-NN?
B

II. What would be the classification of a test sample with x = 4.2 according to 3-NN?
A

III. What is the "leave-one-out" cross validation error of 1-NN. If you need to choose between two or more examples of identical distance, make your choice so that the number of errors is maximized. **[10 Points]**

8 out of 18

II. We have data from a questionnaires survey (to ask people opinion) and objective testing with two attributes(acid durability and strength) to classify whether a special paper tissue is good or not. Here are the four training examples

X1 = Acid durability (in seconda)	X2 = Strength (Kg/sq meter)	Y = Classification
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

Now the factory produces a new tissue that pass laboratory test X1 = 3 and X2 = 7. Without another expensive survey, can we guess the classification of the new tissue using K-nearest neighbor algorithm using k= 3? **[10 Points]**

Distance between data point X1=3 and X2=7 and point 1 = 4

Distance between data point X1=3 and X2=7 and point 2 = 5

Distance between data point X1=3 and X2=7 and point 3 = 3

Distance between data point X1=3 and X2=7 and point 4 = 3.605

So classification using k-nearest neighbor and k= 3 is Good.

III. Draw a neural network that represents the function f(x1; x2; x3) defined below. You can only use two types of units: linear units and sign units. Recall that the linear unit takes as input weights and attribute values and outputs $w_0 + \sum_i w_i x_i$, while the sign unit outputs +1 if $w_0 + \sum_i w_i x_i,$ > 0 and -1 otherwise.

x1	x2	x3	f(x1,x2,x3)
0	0	0	10
0	0	1	-5
0	1	0	-5
0	1	1	10
1	0	0	-5
1	0	1	10
1	1	0	10
1	1	1	10

You have to write down the precise numeric weights (e.g., -1, -0:5, +1, etc.) as well as the precise units used at each hidden and output node.

The simplest solution is to have two hidden layers and one output node. The first hidden layer has three nodes with each unit being a "sign unit", the second layer has one node, which is also a sign unit and the output node is a linear unit.

The three hidden nodes in the first hidden layer represent the following functions respectively:

1. If $\overline{x_1} \wedge \overline{x_2} \wedge x_3$ evaluates to true, output a +1 otherwise output a -1. An example setting of weights is w1 = -1, w2 = -1 and w3 = 1 and w0 = -0.5.

2. If $\overline{x_1} \wedge x_2 \wedge \overline{x_3}$ evaluates to true, output a +1 otherwise output a -1. An example setting of weights is w1 = -1, w2 = 1 and w3 = -1 and w0 = -0:5.

3. If $x_1 \wedge \overline{x_2} \wedge \overline{x_3}$ evaluates to true, output a +1 otherwise output a -1. An example

setting of weights is w1 = 1, w2 = -1 and w3 = -1 and w0 = -0:5.

The input to the hidden unit in the second hidden layer is the output of all hidden nodes in the first hidden layer. It represents the following function:

1. If at least one of the inputs is a +1, the unit outputs a +1, otherwise it outputs a -1. It thus represents an OR node. An example setting of weights is w0 = -0:5 and w1 =w2 = w3 = 1.

The input to the output node is the output of the node in the second hidden layer. It represents the following function

1. If the input is +1 then output -5, otherwise output 10. An example setting of weights is w0 = 5/2 and w1 = -15/2.

(OR)

Another solution have 2 hidden sign units in layer ONE and one linear unit in layer TWO.

Layer ONE:

Hidden Unit 1 :

if all three inputs are 0 then output +1 else output -1 (w1 =-1, w2=-1, w3=-1, w0=1)

Hidden Unit 2:

If at least two of the inputs are 1 then output +1 else output -1 (w1=1, w2=1, w3=1, w0=-1)

Layer TWO:

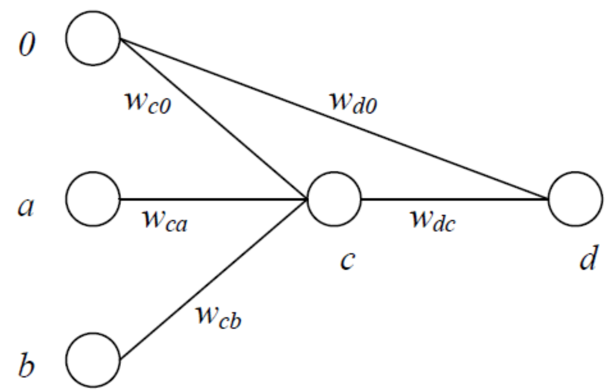
One Linear Unit : if output of one of hidden units in layer One is +1 then output 10 or if both are -1 then -5 (w1=+7.5, w2 =+7.5, w0=+10)

IV. Consider a two-layer feedforward ANN with two inputs a and b, one hidden unit c, and one output unit d. This network has five weights ($w_{ca}, w_{cb}, w_{c0}, w_{dc}, w_{d0}$), where w_{x0} represents the threshold weight for unit x. Initialize these weights to the values (.1,.1,.1,.1,.1), then give their values after each of the first two training iterations of the Backpropagation algorithm. Assume learning rate ɳ =.3, momentum α = 0.9, incremental weight updates, and the following training examples: **[10 Points]**

a	b	d
1	0	1
0	1	0

The network and the sigmoid activation function sigmoid function are as follows:'

$$\sigma(y) = \frac{1}{1+e^{-y}}$$



Training example 1:
The outputs of the two neurons, noting that *a*=1and *b*=0:

$$o_c = \sigma(0.1 * 1 + 0.1 * 0 + 0.1 * 1) = \sigma(0.2) = 0.5498$$
$$o_d = \sigma(0.1 * 0.5498 + 0.1 * 1) = \sigma(0.15498) = 0.53867$$

The error terms for the two neurons, noting that *d*=1:

$$\partial_d = 0.53867 * (1 - 0.53867) * (1 - 0.53867) = 0.1146$$
$$\partial_c = 0.5498 * (1 - 0.5498) * 0.1 * 0.1146 = 0.002836$$

Compute the correction terms as follows, noting that *a*=1, *b*=0 and η = 0.3:

$$\Delta w_{d0} = 0.3 * 0.1146 * 1 = 0.0342$$

$$\Delta w_{dc} = 0.3 * 0.1146 * 0.5498 = 0.0189$$

$$\Delta w_{c0} = 0.3 * 0.002836 * 1 = 0.000849$$

$$\Delta w_{ca} = 0.3 * 0.002836 * 1 = 0.000849$$

$$\Delta w_{cb} = 0.3 * 0.002836 * 0 = 0$$

and the new weights become:

$$w_{d0} = 0.1 + 0.0342 = 0.1342$$

$$w_{dc} = 0.1 + 0.0189 = 0.1189$$

$$w_{c0} = 0.1 + 0.000849 = 0.100849$$

$$w_{ca} = 0.1 + 0.000849 = 0.100849$$

$$w_{cb} = 0.1 + 0 = 0.1$$

Training example 2:

The outputs of the two neurons, noting that *a*=0 and *b*=1:

$$o_c = \sigma(0.100849 * 0 + 0.1 * 1 + 0.100849 * 1) = \sigma(0.200849) = 0.55$$
$$o_d = \sigma(0.1189 * 0.55 + 0.1342 * 1) = \sigma(0.1996) = 0.5497$$

The error terms for the two neurons, noting that *d*=0:

$$\partial_d = 0.5497 * (1 - 0.5497) * (0 - 0.5497) = -0.1361$$
$$\partial_c = 0.55 * (1 - 0.55) * 0.1189 * (-0.1361) = -0.004$$

Compute the correction terms as follows, noting that *a*=0, *b*=1 and η = 0.3 and α =0.9:

$$\Delta w_{d0} = 0.3 * (-0.1361) * 1 + 0.9 * 0.0342 = -0.01$$

$$\Delta w_{dc} = 0.3 * (-0.1361) * 0.55 + 0.9 * 0.0189 = -0.0055$$

$$\Delta w_{c0} = 0.3 * (-.004) * 1 + 0.9 * 0.000849 = -0.0004$$

$$\Delta w_{ca} = 0.3 * (-.004) * 0 + 0.9 * 0.000849 = 0.00086$$

$$\Delta w_{cb} = 0.3 * (-.004) * 1 + 0.9 * 0 = -0.0012$$

and the new weights become:

$$w_{d0} = 0.1342 - .01 = 0.1242$$

$$w_{dc} = 0.1189 - 0.0055 = 0.1134$$

$$w_{c0} = 0.100849 - .0004 = 0.100809$$

$$w_{ca} = 0.100849 + 0.00086 = 0.1016$$

$$w_{cb} = 0.1 - 0.0012 = 0.0988$$

V. Revise the BACKPROPAGATION algorithm in Table 4.2 in Tom Mitchell book so that it operates on units using the squashing function $tanh$ in place of the sigmoid function. That is, assume the output of a single unit is $o = tanh(\vec{w} * \vec{x})$). Give the weight update rule for output layer weights and hidden layer weights. Hint: $tanh'(x) = 1 - tanh^2(x)$.

[10 Points]

Steps T4.3 and T4.4 in Table 4.2 will become as follows, respectively:

$$\partial_k \rightarrow (1 - o_k^2)(t_k - o_k)$$

$$\partial_h \rightarrow (1 - o_h^2) \sum w_{kh} \partial_k$$