Tutorial 5: Neural Networks

1. We have a 5 input, single output single layer network with weights $\mathbf{W}$

$$\mathbf{W} = \begin{bmatrix} 0.1 & 0.2 & 0.1 & 0.15 & 0.05 \end{bmatrix}$$

   and bias = 0.05, where a linear activation function is assumed.

   We have 3 training examples, $(\mathbf{X}, \mathbf{T})$, expressed as a matrix, one row per training item:

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix} \qquad \mathbf{T} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

   What are the weights after one iteration of gradient descent, with learning rate $\eta = 0.01$?

2. Find the first-order derivative of the logistic sigmoid function, and sketch a graph of the derivative. What are the maximum and minimum values of the derivative and when they are achieved?

3. Consider a single-layer network with a single output y with a logistic sigmoid transfer function.

$$y = g(a) = \frac{1}{1 + \exp(-a)}$$

   This is used for a two-class problem with classes C_1 (denoted by target variable $t = 1$) and C_2 (denoted by $t = 0$). We showed in the notes that in the case of a logistic sigmoid transfer function we can interpret y as the conditional probability $P(C_1|\mathbf{x})$ and $(1 - y)$ as $P(C_2|\mathbf{x})$, where $\mathbf{x}$ is the input vector.

   The target t is a binary variable. We know that, given $\mathbf{x}$, the probability of $t = 1$ is $P(t = 1 \mid \mathbf{x}) = P(C_1|\mathbf{x}) = y$; likewise we have $P(t = 0 \mid \mathbf{x}) = P(C_2|\mathbf{x}) = 1 - y$. We can combine this information and write the distribution of the target t in the form:

$$P(t|x, \mathbf{W}) = y^t (1 - y)^{1-t}$$

   This is a Bernoulli distribution, which we met earlier. Note that we have also explicitly shown the dependence on the weights. We can write the log probability:

$$\ln P(t|x, \mathbf{W}) = t \ln y + (1 - t) \ln(1 - y)$$

   We can use this to optimise the weights $\mathbf{W}$ to maximise the log probability—or to minimise the negative log probability. We can do this by writing the error function as follows:

$$E(\mathbf{W}) = -(t \ln y + (1 - t) \ln(1 - y)) \,.$$

   This is the error for a single training example, denoted as E_n in the notes. To avoid clutter the superscript n is ignored in this question

   If we want to train by gradient descent, we need the derivative $\partial E / \partial w_j$ (where w_j connects the jth input to the single output). What is the expression for this derivative? Compare it with the derivative obtained when the sum square error function is used.

Now consider the C class case in which we have the usual "1-from-C" coding scheme, and in which the kth output y_k is interpreted as $P(C_k|\mathbf{x})$. In this case the negative log probability error function is:

$$E(\mathbf{W}) = -\sum_{k=1}^{C} t_k \ln y_k$$

If the transfer function for the output units is the softmax:

$$y_k = \frac{\exp(a_k)}{\sum_{j=1}^{C} \exp(a_j)}$$

then what is the expression for the derivative $\partial E/\partial w_{kj}$ in this case? Comment on your answer. (This is a slightly trickier derivative, remember that a particular y_k depends on all the a_j.)