1 Problem 1

Since

$$t = y(x) + \epsilon$$

Based on Maximun likelihood, we have:

$$\theta = \operatorname*{argmax}_{\theta} P(x, t|\theta)$$

$$\theta = \operatorname*{argmax}_{\theta} \prod_{i=1} P(x_i, t_i|\theta)$$

$$\theta = \operatorname*{argmax}_{\theta} \prod_{i=1} P(t_i|x_i)P(x_i)$$

$P(x_i)$ could be seen as a constant. Then we have:

$$\theta = \operatorname*{argmax}_{\theta} \prod_{i=1} P(t_i|x_i)$$

Given $P(t_i|x_i)$ follows the Gaussian distribution .

$$\theta = \operatorname*{argmax}_{\theta} \prod_{i=1} \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(t_i - f(x_i, \theta))^2}{2\sigma^2})$$

Using Log function, we can obtain:

$$\theta = \operatorname*{argmax}_{\theta} \sum_{i=1} -(t_i - f(x_i, \theta))^2$$

Thus

$$\theta = \operatorname*{argmin}_{\theta} \sum_{i=1} (t_i - f((1, x_i), \theta))^2$$

2 Problem2

2.1 Derivation

(a) For output layer (digest reduction with materials on slides):

$$\delta_k = -\frac{\partial E^n}{\partial a_k^n} = -\frac{\partial E^n}{\partial y_k^n}\frac{\partial y_k^n}{\partial a_k^n} = t_k - y_k$$

(b) For hidden layer (digest reduction with materials on slides):

$$\delta_j = -\frac{\partial E^n}{\partial a_j^n} = -\sum_{k=1}^{c} \frac{\partial E^n}{\partial a_k^n}\frac{\partial a_k^n}{\partial a_j^n}$$

$$= -\sum_{k=1}^{c} \frac{\partial E^n}{\partial a_k^n}\frac{\partial a_k^n}{\partial z_j^n}\frac{\partial z_j^n}{\partial a_j^n}$$

$$= g'(a_j)\sum_{k=1}^{c}(t_k - y_k)\omega_{jk}$$

$$= \sigma(a_j)\sigma(-a_j))\sum_{k=1}^{c}(t_k - y_k)\omega_{jk}$$

2.2 Update rule

It is known that in gradient descent:

$$\omega_i(t+1) = \omega_i(t) - \alpha\frac{\partial E^n}{\partial \omega_i}$$

(a) For output layer:

$$\omega_{jk}(t+1) = \omega_{jk}(t) - \alpha(t_k - y_k)z_j$$

(b)

$$\omega_{ij}(t+1) = \omega_{ij}(t) - \alpha\sigma(a_j)\sigma(-a_j))\sum_{k=1}^{c}(t_k - y_k)\omega_{jk}x_i$$

2.3 Vectorize computation

(a) For hidden layer to output layer:

$$W_{HO} = W_{HO} + \alpha. * (Z^T * (T - Y))$$

(b) For input layer to hidden layer:

$$W_{IH} = W_{IH} + \alpha. * (X^T * (\sigma(a_j). * \sigma(-a_j). * ((T - Y) * \omega^T)))$$

2.4 Classification

i.
Here we use the
By using Z-score, we can obtain the Gaussian distribution of these statistics, and these values are distributed no further than 3 times of the standard deviation. Thus we can avoid some distortion caused by the extreme situation, like very small iris and very large iris of the size as a whole. By using Z-scores of the testing set with respect to the mean and standard deviation, these statistics are more likely to be around the corresponding portion.

ii.
Here are the six 2 dimensional feature spaces for pairs of features.

As is shown in the 6 pictures, it is obvious that we can use a line to separate these two classes. So these classes are linearly separable. As different type of iris should have different features and the values of the two classes can be divided into two portions using a line.

iii.
Here we set the total number of training steps to be 500 as the stopping criterion. And initialize all of the four weights, as well as the threshold to be 0.
And we use perceptron learning algorithm to obtain the weights and threshold.
The source code is in the appendix part.
iv.
The error rate is: 0.
The source code is in the appendix part.

v.
The learning rate I used is 1.
And no matter how I raise or lower the learning rate to be 2, 1, 0.5, .25, the error rate remains 0, the error rate does not change. This is because the training set and the testing set are using Z-score and they are linear separable, and the testing set can be easily calculated to the right class portion.