

## Solutions to Problem set 4

### Problem 1

It is not possible to separate the two classes with a linear decision boundary, see figure 1.

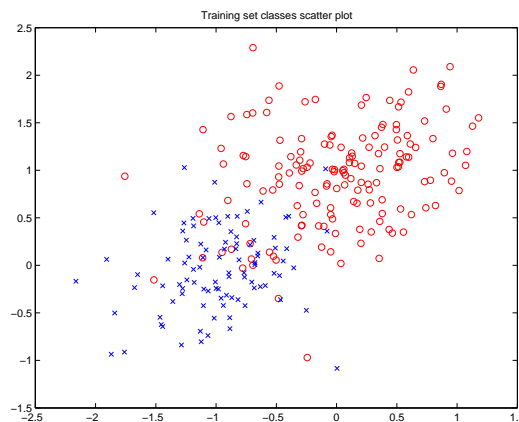


Figure 1: The two classes cannot be separated by linear decision boundary.

### Problem 2

**Part a.** Given the log-likelihood function as:

$$l(D, \mathbf{w}) = \sum_{i=1}^n \mathbf{y}_i \log \mu_i + (1 - \mathbf{y}_i) \log (1 - \mu_i),$$

where  $\mu_i = g(z_i) = g(\mathbf{w}^T \mathbf{x})$  is the logistic function. As is easy to see, the derivative of the logistic function with respect to  $w_i$  is

$$\frac{\partial g(\mathbf{w}^T \mathbf{x})}{\partial w_j} = -\frac{e^{-z} x_j}{(1 + e^{-z})^2} = -x_j g(z)(1 - g(z)).$$

Now we are ready to compute the partial derivative of the log-likelihood function:

$$\frac{\partial l(D, \mathbf{w})}{\partial w_j} = \sum_{i=1}^n \frac{g'(z_i)}{g(z_i)} + (1 - y_i) \frac{g'(z_i)}{(1 - g(z_i))},$$

where

$$g'(z_i) = \frac{\partial g(\mathbf{w}^T \mathbf{x}_i)}{\partial w_j} = -x_{i,j}g(z_i)(1 - g(z_i)).$$

Substituting above expression for  $g'(z_i)$  we obtain:

$$\begin{aligned} \frac{\partial l(D, \mathbf{w})}{\partial w_j} &= \\ & - \sum_{i=1}^n y_i x_{i,j} (1 - g(z_i)) + (1 - y_i) x_{i,j} g(z_i) \\ & = x_{i,j} (y_i - g(z_i)). \end{aligned}$$

The gradient of  $l(D, \mathbf{w})$  is then:

$$\nabla l(D, \mathbf{w}) = \sum_{i=1}^n -\mathbf{x}_i (y_i - \mathbf{g}(\mathbf{z}_i)).$$

**c.**

Weights are:

16.0552

36.2745

34.8291

Confusion matrix for training set is:

79	16
14	141

Confusion matrix for test set is:

29	9
3	59

Mean squared training error is: 0.12.

Mean square test error is: 0.12

**d.** The updated function is main2d.m. See Figure 2.

**e.** See Figures 3-7. All the different settings for  $\alpha$  lead to similar results, probably meaning that the same minimum is reached in all cases.

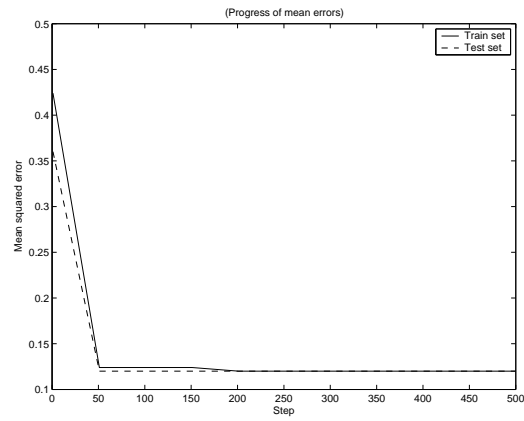


Figure 2: Progress of the errors for 500 steps.

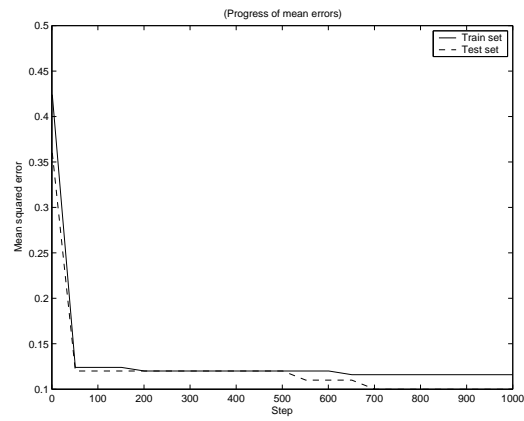


Figure 3: Progress of the errors for  $K = 1000$ ,  $\alpha = 2/k$ .

### Problem 3

b.

Confusion matrix for training set is:

81	14
16	139

Confusion matrix for test set is:

32	6
3	59

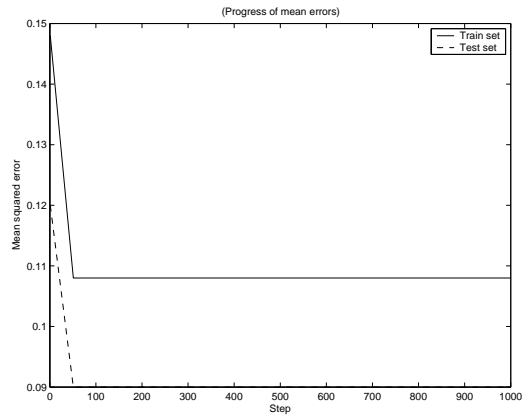


Figure 4: Progress of the errors for  $K = 1000, \alpha = 0.05$ .

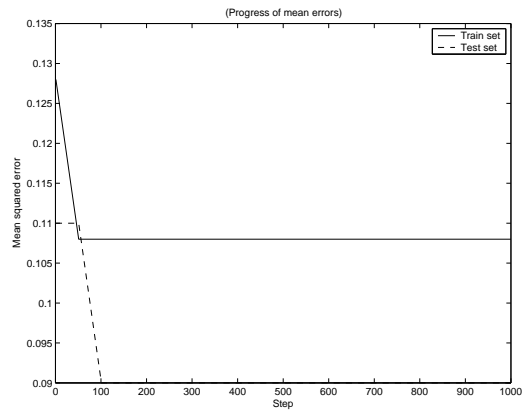


Figure 5: Progress of the errors for  $K = 1000, \alpha = 0.01$ .

Mean squared training error is: 0.12.

Mean square test error is: 0.09.

See Figure 8.

**c.** As can be seen from Figures 9-13, the online gradient method leads to a somewhat better testing error (and much faster!) and about the same performance on the training set as the gradient method of problem 2. The better performance on the test set is also reflected by the mean squared error. This slightly better performance on the test set is reflected in other settings for  $\alpha$  as well.

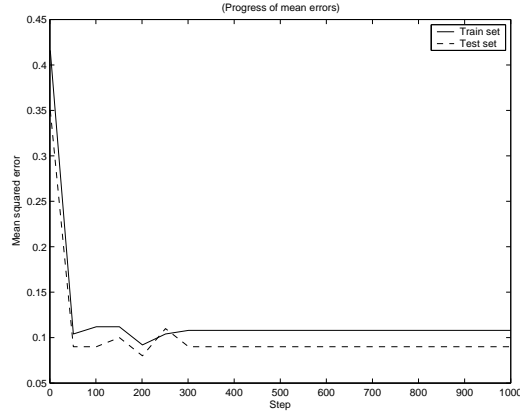


Figure 6: Progress of the errors for  $K = 1000$ ,  $\alpha = 1/\sqrt{k}$ .

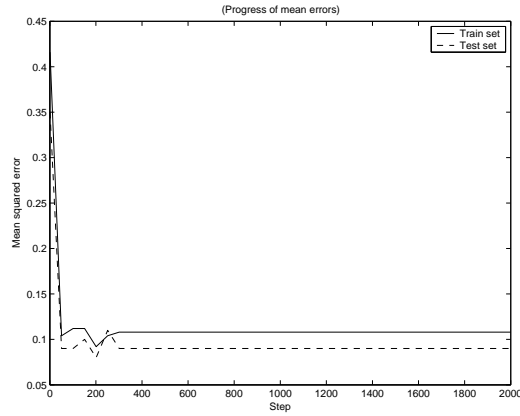


Figure 7: Progress of the errors for  $K = 2000$ ,  $\alpha = 1/\sqrt{k}$ .

**Part a.** The formula is

$$\hat{\mu}_0 = \frac{1}{N} \sum_{i=1, c_i=0}^N x^{(i)} \quad \hat{\mu}_1 = \frac{1}{N} \sum_{i=1, c_i=1}^N x^{(i)}$$

**Part b.** We assume that covariances of both class conditional densities are the same. To estimate the covariances in this case we use a "polled" estimate of covariance matrices:

$$\Sigma = \frac{(N_0) \cdot \Sigma_0 + (N_1) \cdot \Sigma_1}{N_0 + N_1}$$

To obtain the above formula you should realize that covariances measure deviations from the mean. In this case, we have two different means but the same deviations, both class

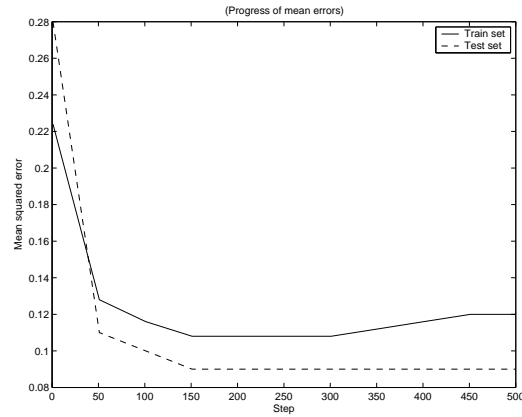


Figure 8: Progress of the errors for  $K = 500$ ,  $\alpha = 2/k$ .

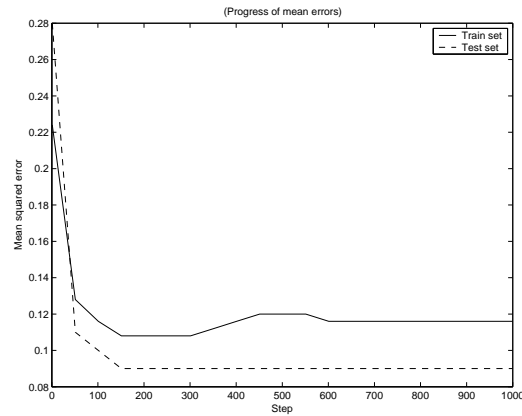


Figure 9: Progress of the errors for  $K = 1000$ ,  $\alpha = 2/k$ .

1 and class 0 examples should count and contribute to the estimate. The whole issue is to weight appropriately covariances from the two classes. Assuming the ML estimates of covariance matrices  $\Sigma_0, \Sigma_1$ , the “polled” formula corresponds to the ML estimate of the parameters in  $\Sigma$ . Formulas for ML estimates of covariance matrices can be found in chapter 3 of Duda, Hart, Stork (Pattern classification).

Note that the ML estimate of the parameters of  $\Sigma$  is biased. You may choose to use also unbiased estimate that is obtained by

$$\Sigma = \frac{(N_0 - 1) \cdot \Sigma_0 + (N_1 - 1) \cdot \Sigma_1}{N_0 + N_1 - 2}$$

where  $\Sigma_0$  and  $\Sigma_1$  are unbiased estimates of covariances of class conditional densities.

**Part c.** Class priors (follow the Bernoulli distribution) can be estimated by computing

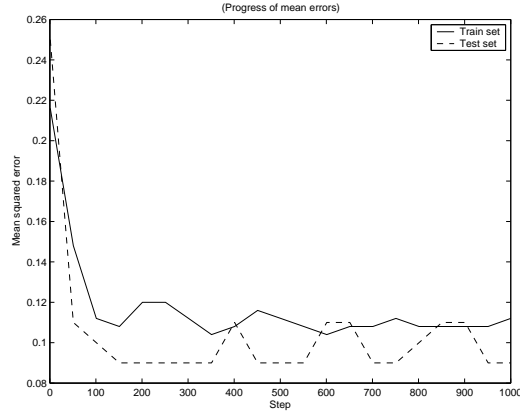


Figure 10: Progress of the errors for  $K = 1000$ ,  $\alpha = 0.05$ .

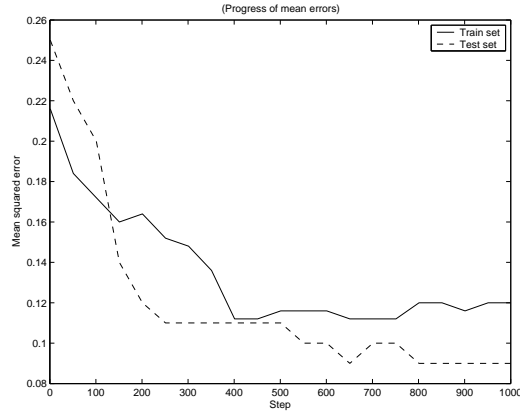


Figure 11: Progress of the errors for  $K = 1000$ ,  $\alpha = 0.01$ .

the number of examples in the training set that belong to each class:  $N_0$  and  $N_1$ . Class probabilities are estimated as:

$$p_0 = \frac{N_0}{N_0 + N_1}$$

$$p_1 = \frac{N_1}{N_0 + N_1}$$

It is easy to convince yourself that this is also the ML estimate. Let  $\theta_{c=1}$  denotes the probability of class 1. Assuming that samples seen in the dataset are independent and identically distributed the likelihood of observing a specific sequence of  $ys$  in the data is:

$$\theta_{c=1}^{N_1} (1 - \theta_{c=1})^{N_0}.$$

In the ML parameter estimation we want to find the value  $\theta_{c=1}$  maximizing the likelihood, i.e.:

$$\theta_{c=1}^{ML} = \arg \max_{\theta_{c=1}} \theta_{c=1}^{N_1} (1 - \theta_{c=1})^{N_0}.$$

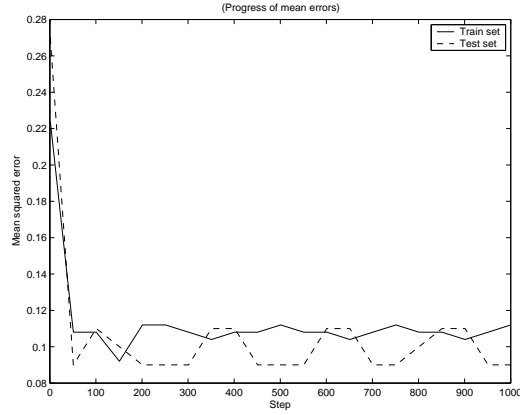


Figure 12: Progress of the errors for  $K = 1000$ ,  $\alpha = 1/\sqrt{k}$ .

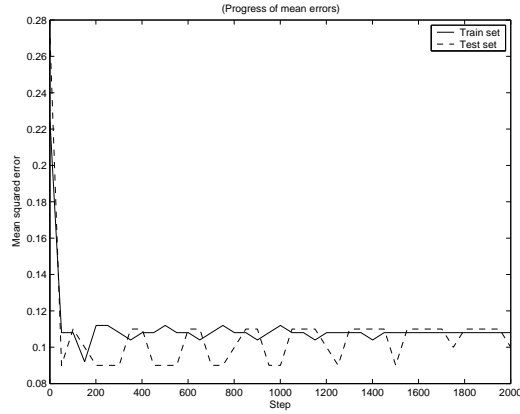


Figure 13: Progress of the errors for  $K = 2000$ ,  $\alpha = 1/\sqrt{k}$ .

This quantity can be optimized by optimizing the loglikelihood of data.

$$l(D, \theta) = N_1 \log \theta_{c=1} + N_0 \log(1 - \theta_{c=1})$$

Setting the derivative of the loglikelihood equal to zero we obtain:

$$\theta_{c=1}^{ML} = \frac{N_0}{N_0 + N_1}.$$

By substituting actual counts to the expression we get:  $p_{c=1} = \theta_{c=1} = 0.62$  and  $p_{c=0} = (1 - \theta_{c=1}) = 0.38$ .

Once the parameters of the generative model are estimated we can use the model for making predictions about class labels for inputs  $\mathbf{x}$ . To choose the class we examine posterior probabilities for both classes and choose the one with the highest posterior probability. Applying

the posterior choice to our datasets we got the following mean misclassification errors:

$$\begin{aligned} MME_{Train} &= 0.100 \\ MME_{Test} &= 0.100 \end{aligned}$$

**Parts d,e.** The discriminant functions for the posterior choice model and non-equal covariance matrices are:

$$\begin{aligned} g_0(\mathbf{x}) &= \\ &-\frac{1}{2}(\mathbf{x} - \mu_0)^T \cdot \Sigma_0^{-1} \cdot (\mathbf{x} - \mu_0) - \frac{1}{2}\log(|\Sigma_0|) + \log(p_0) \\ g_1(\mathbf{x}) &= \\ &-\frac{1}{2}(\mathbf{x} - \mu_1)^T \cdot \Sigma_1^{-1} \cdot (\mathbf{x} - \mu_1) - \frac{1}{2}\log(|\Sigma_1|) + \log(p_1); \end{aligned}$$

When covariance matrices are equal (our case), discriminant functions become simpler (see chapter 2 of Duda, Hart, Stork):

$$\begin{aligned} g_0(\mathbf{x}) &= (\Sigma_0^{-1}\mu_0)^T \mathbf{x} - \frac{1}{2}\mu_0^T \Sigma_0^{-1} \mu_0 + \log(p_0); \\ g_1(x) &= (\Sigma_1^{-1}\mu_1)^T \mathbf{x} - \frac{1}{2}\mu_1^T \Sigma_1^{-1} \mu_1 + \log(p_1); \end{aligned}$$

**Part g.**

$$\begin{aligned} \mu_0 &= \begin{bmatrix} -0.97663 & -0.043558 \end{bmatrix}^T \\ \mu_1 &= \begin{bmatrix} 0.01564 & 0.94156 \end{bmatrix}^T \end{aligned}$$

$$\Sigma = \begin{pmatrix} 0.2688 & 0.0730 \\ 0.0730 & 0.2470 \end{pmatrix}$$

$$\theta_0 = 0.3800$$

$$\theta_1 = 0.6200$$

Mean squared training error is 0.1.

Mean squared test error is 0.1.

Confusion matrix for training set is:

$$\begin{array}{cc} 87 & 8 \\ 17 & 138 \end{array}$$

Confusion matrix for test set is:

$$\begin{array}{cc} 33 & 5 \\ 5 & 57 \end{array}$$

Withing the variation given by the randomized nature of online gradient descent, the results are comparable and neither model is superior to the other. (In fact, it can be shown they are mathematically equivalent.)