Optimal Classifier

Suppose $K = 2$.

Let the class densities be Gaussian that differ only in their means $\mathbf{m}_1$ and $\mathbf{m}_0$; let the covariance matrix be $\mathbf{\Sigma}$.

Let $\mathbf{w} = \mathbf{\Sigma}^{-T}\left(\mathbf{m}_1 - \mathbf{m}_0\right)$ and $w_{\mathrm{p}} = -\left(\mathbf{m}_1 - \mathbf{m}_0\right)^T \mathbf{\Sigma}^{-1} \dfrac{\left(\mathbf{m}_1 + \mathbf{m}_0\right)}{2} + \log \dfrac{\pi_0}{\pi_1}$.

Define $g\left(\mathbf{x}\right) = \mathbf{w}^T \mathbf{x} + w_p$. The hyperplane $g\left(\mathbf{x}\right) = 0$ is the decision boundary.

The optimal decision rule is to choose $l = 0$ if $g\left(\mathbf{x}\right) < 0$ and $l = 1$ if $g\left(\mathbf{x}\right) > 0$.

What if we do not know $\mathbf{m}_1$, $\mathbf{m}_0$, $\mathbf{\Sigma}$, π_0, or π_1?

Do you know

(i) if the two class densities are Gaussian and

(ii) that the class densities differ only in their class means?

In practice

What if we do not know $\mathbf{m}_1$, $\mathbf{m}_0$, $\mathbf{\Sigma}$, π_0, or π_1 ?

Do you know

(i) if the two class densities are Gaussian and

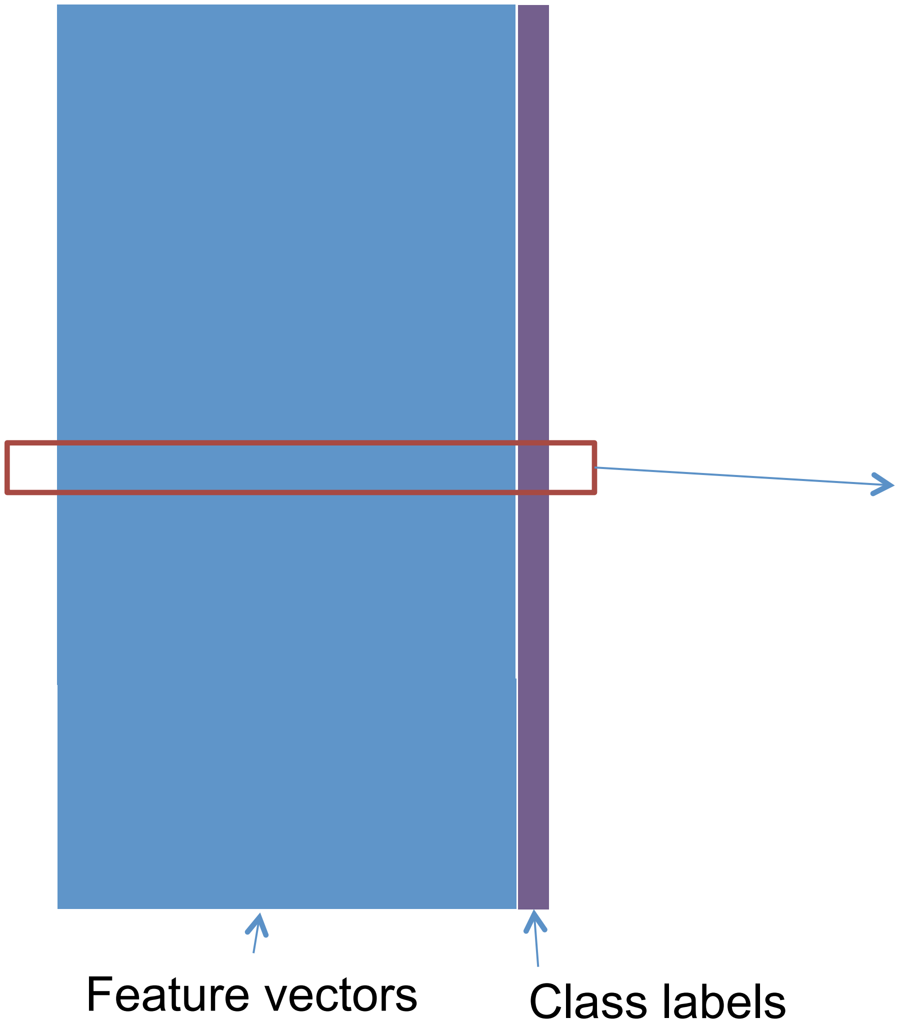
(ii) that the class densities differ only in their class means?

Do you have a data set of labeled data samples?

If *yes* to all of the above, then supervised learning is a solution

Set of Labeled Feature Vectors

Given set of labeled feature vectors



Each row is a set of features and a corresponding class label

$$\left(x_0{}^{(i)}, x_1{}^{(i)}, \cdots, x_{p-1}{}^{(i)}, t^{(i)}\right)$$

Plug-in Classifier

Suppose $K = 2$.

Let the class densities be Gaussian that differ only in their means $\mathbf{m}_1$ and $\mathbf{m}_0$;

let the covariance matrix be $\mathbf{\Sigma}$.

If we know these assumptions are valid, then estimate the unknown parameters from the labeled data set and plug them into the classifier

Let $\mathbf{w} = \mathbf{\Sigma}^{-T}(\mathbf{m}_1 - \mathbf{m}_0)$ and $w_\mathrm{p} = -(\mathbf{m}_1 - \mathbf{m}_0)^T \mathbf{\Sigma}^{-1} \dfrac{(\mathbf{m}_1 + \mathbf{m}_0)}{2} + \log \dfrac{\pi_0}{\pi_1}$.

Define $g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_p$. The hyperplane $g(\mathbf{x}) = 0$ is the decision boundary.

The optimal decision rule is to choose $l = 0$ if $g(\mathbf{x}) < 0$ and $l = 1$ if $g(\mathbf{x}) > 0$.

Estimate the means, covariance matrix, and prior probabilities

Let T be a training data set: $T = \left\{ \left(\mathbf{x}^{(i)}, t^{(i)} \right) : i = 0, \cdots, N-1 \right\}$.

Each training sample pair has an observed feature vector $\mathbf{x}$ and a corresponding target (label) t.

The target $t \in \{0, 1, \cdots, K-1\}$.

When $K = 2$, if we let $t \in \{0, 1\}$, we can interpret the value of t as the probability that $\mathbf{x}$ belongs to Class 1.

When $K > 2$, it is sometimes useful to use a 1-of-K coding so that the training sample pair is $(\mathbf{x}, \mathbf{t})$, where $\mathbf{t}$ is a $K \times 1$ target vector such that if the true class is k, then $t_k = 1$ and $t_{k'} = 0$, for all $k' \neq k$.

This way, the value of t_k can be interpreted as the probability that $\mathbf{x}$ belongs to Class k.

In this case, the training data set is $T = \left\{ \left(\mathbf{x}^{(i)}, \mathbf{t}^{(i)} \right) : i = 0, \cdots, N-1 \right\}$.

Estimate the means, covariance matrix, and prior probabilities

Suppose $K = 2$ and let T be a training data set : $T = \left\{\left(\mathbf{x}^{(i)}, t^{(i)}\right) : i = 0, \cdots, N-1\right\}$.
The target $t \in \{0,1\}$.
Let X_0 and X_1 be defined such that X_k is the set of all vectors labeled Class k, $k = 0,1$,
and $N_k = \left|X_k\right|$ = number of all vectors with label k.

The estimates of the prior probabilities are :

$$\hat{\pi}_0 = \frac{N_0}{N} \text{ and } \hat{\pi}_1 = \frac{N_1}{N}.$$

The estimates of the mean vectors are :

$$\hat{\mathbf{m}}_0 = \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \mathbf{x}^{(i)} \text{ and } \hat{\mathbf{m}}_1 = \frac{1}{N_1} \sum_{\mathbf{x}^{(i)} \in X_1} \mathbf{x}^{(i)}.$$

Estimate the means, covariance matrix, and prior probabilities

Suppose $K = 2$ and let T be a training data set : $T = \left\{\left(\mathbf{x}^{(i)}, t^{(i)}\right): i = 0, \cdots, N-1\right\}$.
The target $t \in \{0,1\}$.
Let X_0 and X_1 be defined such that X_k is the set of all vectors labeled Class k, $k = 0,1$,
and $N_k = \left|X_k\right|$ = number of all vectors with label k.

The estimate of the covariance matrix is :

$$\hat{\mathbf{\Sigma}} = \hat{\pi}_0 \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)^T + \hat{\pi}_1 \frac{1}{N_1} \sum_{\mathbf{x}^{(i)} \in X_1} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)^T$$

$$= \frac{1}{N} \left\{ \sum_{\mathbf{x}^{(i)} \in X_0} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)^T + \sum_{\mathbf{x}^{(i)} \in X_1} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)^T \right\}.$$

Estimate the means, covariance matrix, and prior probabilities

Suppose $K = 2$ and let T be a training data set : $T = \{(\mathbf{x}^{(i)}, t^{(i)}) : i = 0, \cdots, N-1\}$.
The target $t \in \{0,1\}$.
Let X_0 and X_1 be defined such that X_k is the set of all vectors labeled Class k, $k = 0,1$,
and $N_k = |X_k|$ = number of all vectors with label k.

The estimates of the prior probabilities are :

$$\hat{\pi}_0 = \frac{N_0}{N} \text{ and } \hat{\pi}_1 = \frac{N_1}{N}.$$

The estimates of the mean vectors are :

$$\hat{\mathbf{m}}_0 = \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \mathbf{x}^{(i)} \text{ and } \hat{\mathbf{m}}_1 = \frac{1}{N_1} \sum_{\mathbf{x}^{(i)} \in X_1} \mathbf{x}^{(i)}.$$

Estimate the means, covariance matrix, and prior probabilities

Suppose $K = 2$ and let T be a training data set : $T = \{(\mathbf{x}^{(i)}, t^{(i)}) : i = 0, \cdots, N-1\}$.
The target $t \in \{0,1\}$.
Let X_0 and X_1 be defined such that X_k is the set of all vectors labeled Class k, $k = 0,1$,
and $N_k = |X_k|$ = number of all vectors with label k.

The estimate of the covariance matrix is :

$$\hat{\mathbf{\Sigma}} = \hat{\pi}_0 \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)^T + \hat{\pi}_1 \frac{1}{N_1} \sum_{\mathbf{x}^{(i)} \in X_1} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)^T$$

$$= \frac{1}{N} \left\{ \sum_{\mathbf{x}^{(i)} \in X_0} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)^T + \sum_{\mathbf{x}^{(i)} \in X_1} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)^T \right\}.$$

The estimate of the covariance matrix is:

$$\hat{\mathbf{\Sigma}} = \hat{\pi}_0 \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)^T + \hat{\pi}_1 \frac{1}{N_1} \sum_{\mathbf{x}^{(i)} \in X_1} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)^T$$

Issues to consider when we implement this in software

- representation of the variables
- efficiency

Let $\hat{\mathbf{\Sigma}}_0 = \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)^T$.

$$\hat{\mathbf{\Sigma}}_0 = \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left\{\mathbf{x}^{(i)}\mathbf{x}^{(i)T} - \mathbf{x}^{(i)}\hat{\mathbf{m}}_0{}^T - \hat{\mathbf{m}}_0\mathbf{x}^{(i)T} + \hat{\mathbf{m}}_0\hat{\mathbf{m}}_0{}^T\right\}$$

$$= \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left\{\mathbf{x}^{(i)}\mathbf{x}^{(i)T}\right\} - \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left\{\mathbf{x}^{(i)}\hat{\mathbf{m}}_0{}^T\right\} - \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left\{\hat{\mathbf{m}}_0\mathbf{x}^{(i)T}\right\} + \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left\{\hat{\mathbf{m}}_0\hat{\mathbf{m}}_0{}^T\right\}$$

$$= \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left\{\mathbf{x}^{(i)}\mathbf{x}^{(i)T}\right\} - \left(\frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \mathbf{x}^{(i)}\right)\hat{\mathbf{m}}_0{}^T - \hat{\mathbf{m}}_0 \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \mathbf{x}^{(i)T} + \hat{\mathbf{m}}_0\hat{\mathbf{m}}_0{}^T \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} 1$$

$$= \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left\{\mathbf{x}^{(i)}\mathbf{x}^{(i)T}\right\} - \hat{\mathbf{m}}_0\hat{\mathbf{m}}_0{}^T - \hat{\mathbf{m}}_0\hat{\mathbf{m}}_0{}^T + \hat{\mathbf{m}}_0\hat{\mathbf{m}}_0{}^T$$

$$= \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left\{\mathbf{x}^{(i)}\mathbf{x}^{(i)T}\right\} - \hat{\mathbf{m}}_0\hat{\mathbf{m}}_0{}^T.$$

The estimate of the covariance matrix is:

$$\hat{\boldsymbol{\Sigma}} = \hat{\pi}_0 \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_0\right)^T + \hat{\pi}_1 \frac{1}{N_1} \sum_{\mathbf{x}^{(i)} \in X_1} \left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)\left(\mathbf{x}^{(i)} - \hat{\mathbf{m}}_1\right)^T$$

The estimate of the covariance matrix is then:

$$\hat{\boldsymbol{\Sigma}} = \hat{\pi}_0 \hat{\boldsymbol{\Sigma}}_0 + \hat{\pi}_1 \hat{\boldsymbol{\Sigma}}_1$$

$$= \frac{1}{N} \left\{ \sum_{\mathbf{x}^{(i)} \in X_0} \mathbf{x}^{(i)} \mathbf{x}^{(i)T} + \sum_{\mathbf{x}^{(i)} \in X_1} \mathbf{x}^{(i)} \mathbf{x}^{(i)T} \right\} - \hat{\pi}_0 \hat{\mathbf{m}}_0 \hat{\mathbf{m}}_0{}^T - \hat{\pi}_1 \hat{\mathbf{m}}_1 \hat{\mathbf{m}}_1{}^T$$

$$= \frac{1}{N} \sum_{\text{all } \mathbf{x}^{(i)}} \mathbf{x}^{(i)} \mathbf{x}^{(i)T} - \hat{\pi}_0 \hat{\mathbf{m}}_0 \hat{\mathbf{m}}_0{}^T - \hat{\pi}_1 \hat{\mathbf{m}}_1 \hat{\mathbf{m}}_1{}^T$$

How does one compute $\mathbf{S} = \sum_{\text{all } \mathbf{x}^{(i)}} \mathbf{x}^{(i)} \mathbf{x}^{(i)^T}$?

The dimension of $\mathbf{S}$ is $p \times p$.

It is a sum of $p \times p$ matrices $\mathbf{x}^{(i)} \mathbf{x}^{(i)^T}, i = 0, \cdots, N-1$.

The ith term in the sum is $\mathbf{x}^{(i)} \mathbf{x}^{(i)^T} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ \vdots \\ x_{p-1}^{(i)} \end{bmatrix} \begin{bmatrix} x_0^{(i)} & x_1^{(i)} & \cdots & x_{p-1}^{(i)} \end{bmatrix}$

$$= \begin{bmatrix} x_0^{(i)} x_0^{(i)} & x_0^{(i)} x_1^{(i)} & \cdots & x_0^{(i)} x_{p-1}^{(i)} \\ x_1^{(i)} x_0^{(i)} & x_1^{(i)} x_1^{(i)} & \cdots & x_1^{(i)} x_{p-1}^{(i)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{p-1}^{(i)} x_0^{(i)} & x_{p-1}^{(i)} x_1^{(i)} & \cdots & x_{p-1}^{(i)} x_{p-1}^{(i)} \end{bmatrix}.$$

How does one compute $\mathbf{S} = \sum_{\text{all } \mathbf{x}^{(i)}} \mathbf{x}^{(i)} \mathbf{x}^{(i)^T}$?

$$\mathbf{S} = \sum_{i=0}^{N-1} \begin{bmatrix} x_0^{(i)} x_0^{(i)} & x_0^{(i)} x_1^{(i)} & \cdots & x_0^{(i)} x_{p-1}^{(i)} \\ x_1^{(i)} x_0^{(i)} & x_1^{(i)} x_1^{(i)} & \cdots & x_1^{(i)} x_{p-1}^{(i)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{p-1}^{(i)} x_0^{(i)} & x_{p-1}^{(i)} x_1^{(i)} & \cdots & x_{p-1}^{(i)} x_{p-1}^{(i)} \end{bmatrix}.$$

The (r, c)th element of the matrix $\mathbf{S}$ (i.e., the element at row r and column c) is:

$$s_{rc} = \sum_{i=0}^{N-1} x_r^{(i)} x_c^{(i)}.$$

The class average vector $\hat{\mathbf{m}}_0$ is computed as $\hat{\mathbf{m}}_0 = \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} \mathbf{x}^{(i)}$.

The rth element of the vector $\hat{\mathbf{m}}_0$ is:

$$(m_0)_r = \frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} x_r^{(i)};$$

i.e., it is the average of the rth feature over all feature vectors in Class 0.

Similarly, the rth element of the vector $\hat{\mathbf{m}}_1$ is:

$$(m_1)_r = \frac{1}{N_1} \sum_{\mathbf{x}^{(i)} \in X_1} x_r^{(i)}.$$

What is $\mathbf{M}_0 = \hat{\mathbf{m}}_0 \hat{\mathbf{m}}_0{}^T$?

The matrix $\mathbf{M}_0$ is

$$\mathbf{M}_0 = \begin{bmatrix} (m_0)_0 \\ (m_0)_1 \\ \vdots \\ (m_0)_{p-1} \end{bmatrix} \begin{bmatrix} (m_0)_0 & (m_0)_1 & \cdots & (m_0)_{p-1} \end{bmatrix}$$

$$= \begin{bmatrix} (m_0)_0 (m_0)_0 & (m_0)_0 (m_0)_1 & \cdots & (m_0)_0 (m_0)_{p-1} \\ (m_0)_1 (m_0)_0 & (m_0)_1 (m_0)_1 & \cdots & \\ \vdots & \vdots & \ddots & \\ (m_0)_{p-1} (m_0)_0 & & & (m_0)_{p-1} (m_0)_{p-1} \end{bmatrix}.$$

The (r,c)th element of the matrix $\mathbf{M}_0$ (i.e., the element at row r and column c) is :

$$(m_0)_{rc} = (m_0)_r (m_0)_c = \left(\frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} x_r{}^{(i)} \right) \left(\frac{1}{N_0} \sum_{\mathbf{x}^{(i)} \in X_0} x_c{}^{(i)} \right).$$

Note that the product of two sums does not equal the sum of two products!

The estimate of the covariance matrix is:

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{N} \sum_{\text{all } \mathbf{x}^{(i)}} \mathbf{x}^{(i)} \mathbf{x}^{(i)^T} - \hat{\pi}_0 \hat{\mathbf{m}}_0 \hat{\mathbf{m}}_0{}^T - \hat{\pi}_1 \hat{\mathbf{m}}_1 \hat{\mathbf{m}}_1{}^T .$$

The (r,c)th element of $\hat{\boldsymbol{\Sigma}}$ is:

$$\sigma_{rc} = \frac{1}{N} \sum_{i=0}^{N-1} x_r^{(i)} x_c^{(i)} - \frac{N_0}{N} (m_0)_r (m_0)_c - \frac{N_1}{N} (m_1)_r (m_1)_c .$$

Given a set of labeled vectors $\left\{\left(x_0^{(i)}, x_1^{(i)}, \cdots, x_{p-1}^{(i)}, t^{(i)}\right) : i = 0, \cdots N-1\right\}$,

we want to compute the prior probabilities for the two classes.

Define scalar variables `pi0`, `pi1` for the prior probabilities.

Initialize all variables to zero.

```
for (i=0; i<N; i++){  // for each sample vector
        if (t[i] == 0) // label is class 0
                pi0++;
        else // label is class 1
                pi1++;
}

pi0 /= N;  // estimate of prior probability for class 0
pi1 /= N;  // estimate of prior probability for class 1
```

Assumes the labels are stored as an array: one label per row

Given a set of labeled vectors $\left\{\left(x_0{}^{(i)}, x_1{}^{(i)}, \cdots, x_{p-1}{}^{(i)}, t^{(i)}\right): i = 0, \cdots N-1\right\}$, we want to compute the class averages for the two classes.

Define array variables `m0`, `m1` for the class averages and counters `count0, count1` for the classes

Initialize all variables and counters to zero.

```
for (i=0; i<N; i++){  // for each sample vector
        if (t[i] == 0) { // label is class 0
                count0++;
                // loop over all features
                for (j=0; j<p; j++)
                        m0[j] += x[i][j];
        }
        else { // label is class 1
                count1++;
                for (j=0; j<p; j++)
                        m1[j] += x[i][j];
        }
}

// divide by the counters to compute the class averages
for (j=0; j<p; j++) {  // for each feature
        m0[j] /= count0;
        m1[j] /= count1;
}
```

Assumes the sample vectors are stored as a 2D array: one vector per row; each column is a feature

Given a set of labeled vectors $\left\{\left(x_0^{(i)}, x_1^{(i)}, \cdots, x_{p-1}^{(i)}, t^{(i)}\right) : i = 0, \cdots N-1\right\}$,

we want to compute the prior probabilities and the class averages for the plug - in classifier.

Define scalar variables `pi0`, `pi1` as the prior probabilities, array variables `m0`, `m1` as the class averages, and `count0`, `count1` as counters for the classes

Initialize all variables and counters to zero.

```
for (i=0; i<N; i++){   // for each sample vector
        if (t[i] == 0) { // label is class 0
                count0++;
                for (j=0; j<p; j++) // loop over all features
                        m0[j] += x[i][j];
        }
        else { // label is class 1
                count1++;
                for (j=0; j<p; j++)
                        m1[j] += x[i][j];
        }
}

// divide by the counters to compute the class averages
for (j=0; j<p; j++) {   // for each feature
        m0[j] /= count0;
        m1[j] /= count1;
}

// divide counters by total count to compute the prior probabilities
pi0 = count0/N;
pi1 = count1/N;
```

Given a set of labeled vectors $\left\{\left(x_0^{(i)}, x_1^{(i)}, \cdots, x_{p-1}^{(i)}, t^{(i)}\right): i = 0, \cdots N-1\right\}$,

we want to compute the covariance matrix for the plug - in classifier.

Assume scalar variables pi0, pi1 hold the prior probabilities, array variables `m0`, `m1` hold the class averages.
Define 2d array `cov` for the covariance matrix

The (r,c)th element of $\hat{\Sigma}$ is:

$$\sigma_{rc} = \frac{1}{N}\sum_{i=0}^{N-1} x_r^{(i)} x_c^{(i)} - \frac{N_0}{N}(m_0)_r (m_0)_c - \frac{N_1}{N}(m_1)_r (m_1)_c.$$

Initialize `cov` to zero.

```
// compute the covariance matrix element by element
for (row=0; row<p; row++) for (col=0; col<p; col++) {
        // for each correlation term, sum over all vectors
        for (i=0; i<N; i++) {
                cov[row][col] += x[i][row]*x[i][col];
        }

        // divide by N to get the first term
        cov[row][col] /= N;

        // subtract the 2nd term
        cov[row][col] -= pi0*m0[row]*m0[col];

        // subtract the 3rd term
        cov[row][col] -= pi1*m1[row]*m1[col];
}
```

Do we need to compute all p^2 elements?

Can we combine this with the computations of the prior probabilities and class averages?

Given a set of labeled vectors $\left\{\left(x_0^{(i)}, x_1^{(i)}, \cdots, x_{p-1}^{(i)}, t^{(i)}\right): i = 0, \cdots N-1\right\}$,

we want to compute the covariance matrix for the plug-in classifier.

Assume scalar variables pi0, pi1 hold the prior probabilities, array variables `m0`, `m1` hold the class averages.
Define 2d array `cov` for the covariance matrix

The (r,c)th element of $\hat{\Sigma}$ is:

$$\sigma_{rc} = \frac{1}{N}\sum_{i=0}^{N-1} x_r^{(i)} x_c^{(i)} - \frac{N_0}{N}(m_0)_r (m_0)_c - \frac{N_1}{N}(m_1)_r (m_1)_c.$$

Initialize `cov` to zero.

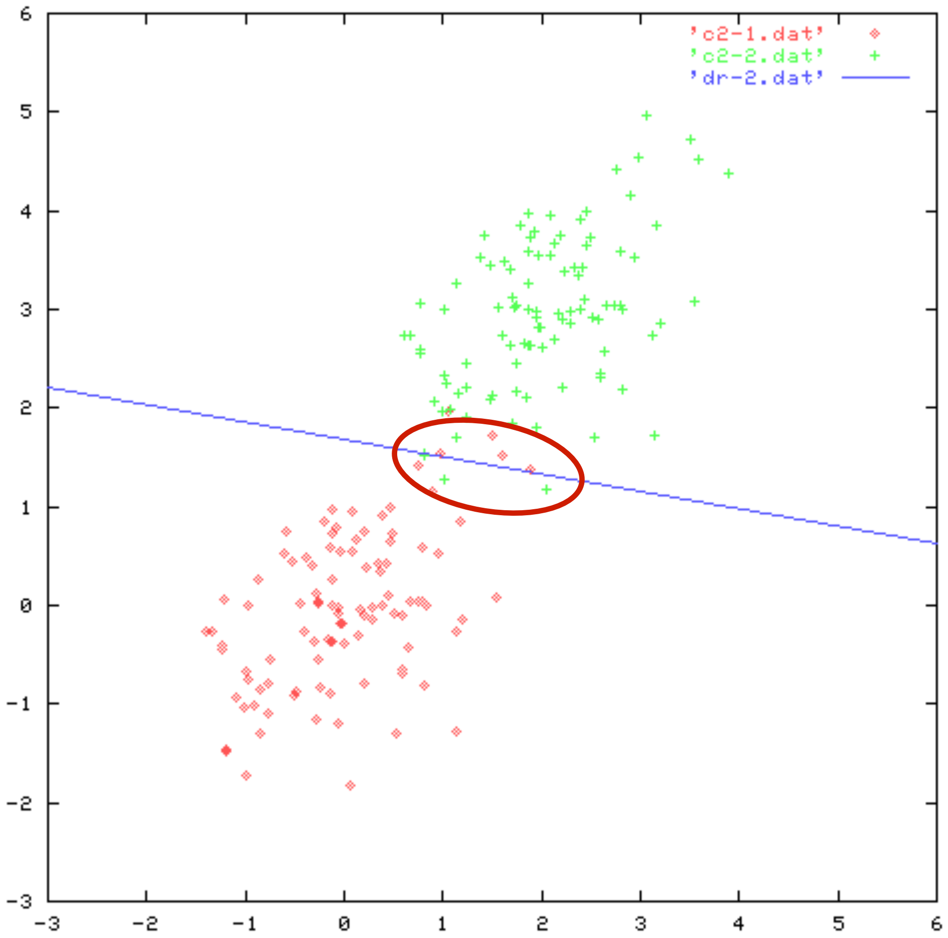
```
// compute the covariance matrix element by element
for (row=0; row<p; row++) for (col=0; col<=row; col++) {
        // for each correlation term, sum over all vectors
        for (i=0; i<N; i++) {
                cov[row][col] += x[i][row]*x[i][col];
        }

        // divide by N to get the first term
        cov[row][col] /= N;

        // subtract the 2nd term
        cov[row][col] -= pi0*m0[row]*m0[col];

        // subtract the 3rd term
        cov[row][col] -= pi1*m1[row]*m1[col];

        // do the lower triangle term
        cov[col][row] = cov[row][col];
}
```

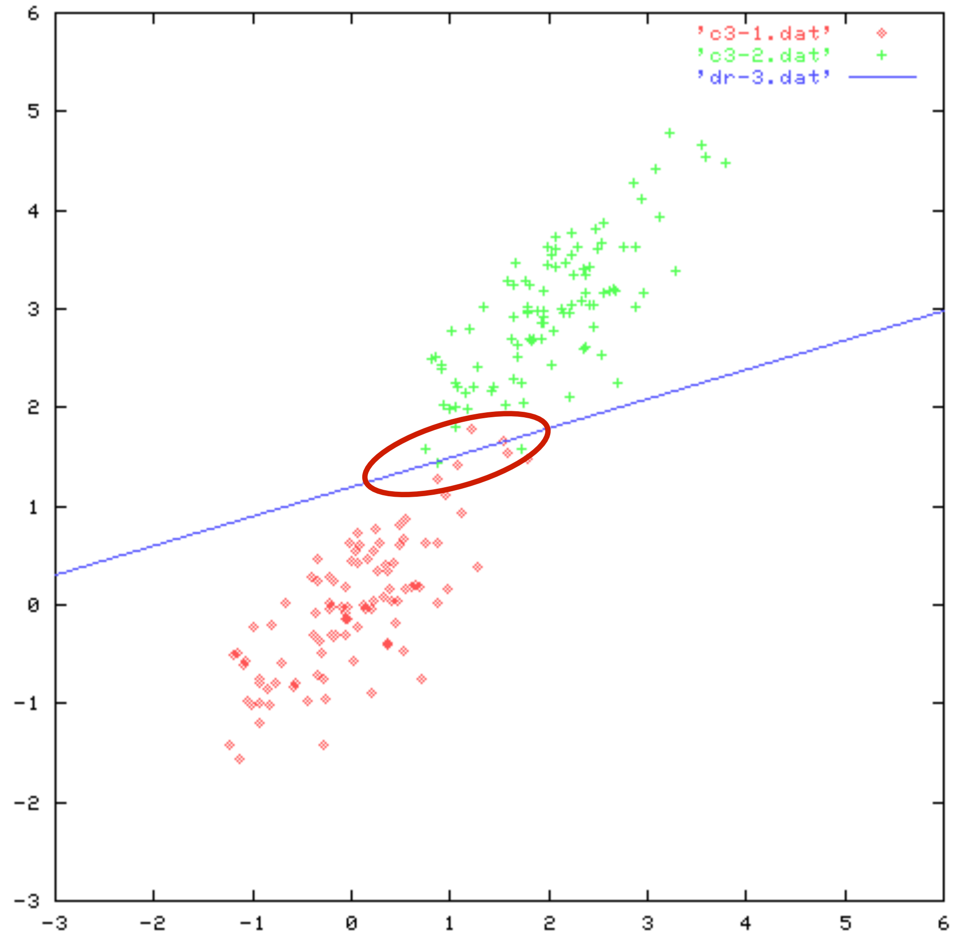
We do not need to compute all p^2 elements because the covariance matrix is symmetric

How do we measure performance?

- Use the plugged in values and solve for the expected probability of misclassification
- Or, since we have a training data set, count the misclassifications

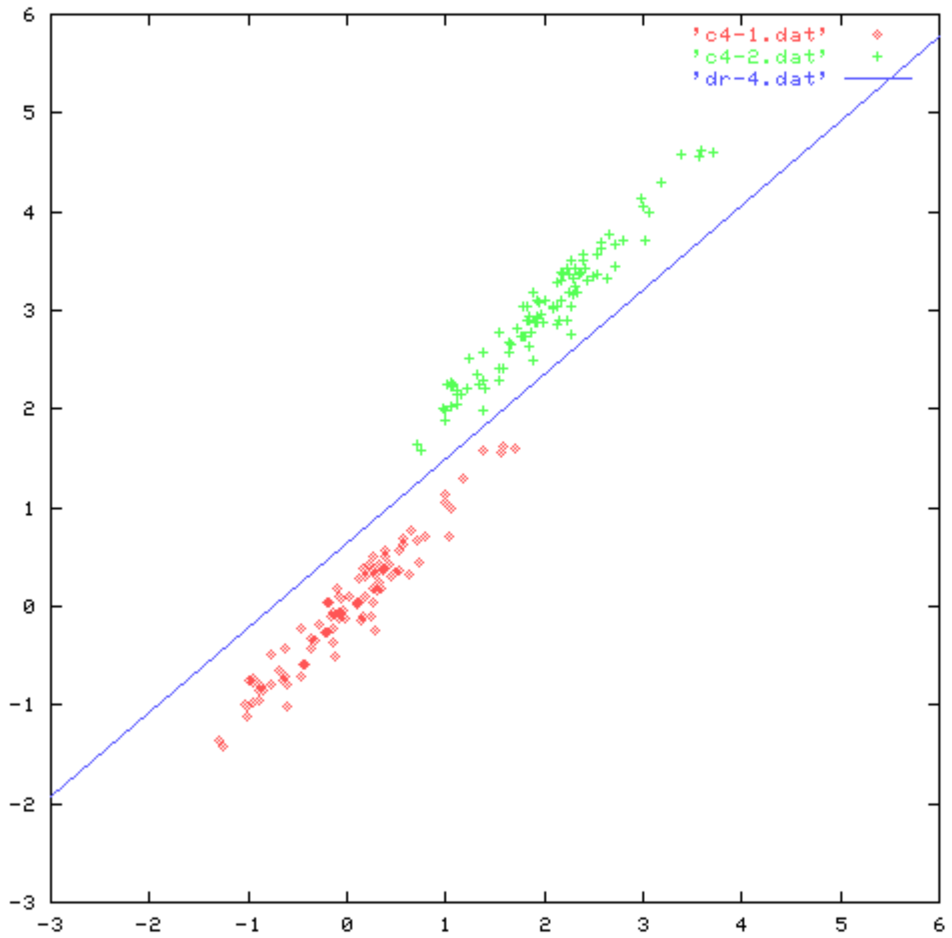
Misclassifications





Misclassification rate = #misclassified / #training samples

Misclassifications



Misclassification rate = 0?

Fitting of a decision boundary

- Overfitting (relative to the training set)
  - Can quote a low misclassification rate
  - Unknown performance when dealing with a different data set (*ie* when the classifier has to be in operating condition)
- Underfitting
  - Higher misclassification rate relative to the training set
  - High misclassification rate when dealing with a different data set (though it may be better than an overfitted classifier)

Honest evaluation of misclassification rate

- Partition a given labeled sample set into a training set and a test set
- Optimize the fitting of the decision boundary using the training set
- Quote the performance of the trained classifier by the misclassification rate estimated using the test set

Partitioning the data set

Strategies

- Equally divide into training and test sets
- Make sure the two sets have similar distributions (so that one set does not have all samples from a particular class)
- In general, the assumption is that the data set (training or test) should have similar distributions to the actual samples

Partitioning the data set

- Problem if the observed data set is not sufficiently large or balanced

  Eg: N=340, N_0=337, N_1=3

Cross Validation

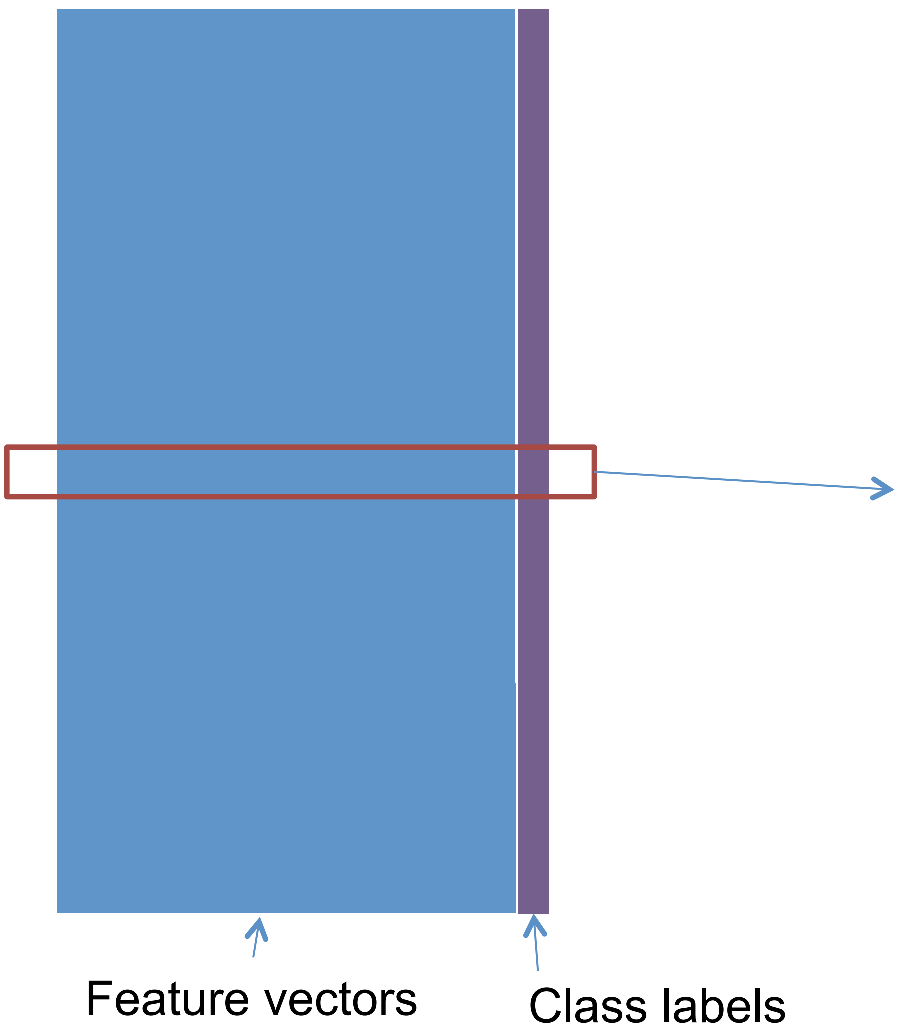
- Partition the data set T into *P* partitions: $T_0, T_1, \ldots, T_{P-1}$
- For each *k*, *k*=0,…,*P*-1,
  - Use T_k as the test set
  - Use the other (*P*-1) partitions as the training set
- Overall performance is the average (over the *P* runs) misclassification rate
- Smallest *P* is 2; the largest *P* is *N*
- When *P=N*, it is called "leave-one-out" strategy

Cross Validation

- Partition the data set T into P partitions:
  $T_0, T_1, \ldots, T_{P-1}$
- For each k, $k=0,\ldots,P-1$,
  - Use T_k as the test set
  - Use the other (P-1) partitions as the training set
- Overall performance is the average (over the P runs) misclassification rate
- Smallest P is 2; the largest P is N
- When $P=N$, it is called "leave-one-out" strategy

Labeled Feature Vectors

Given set of labeled feature vectors

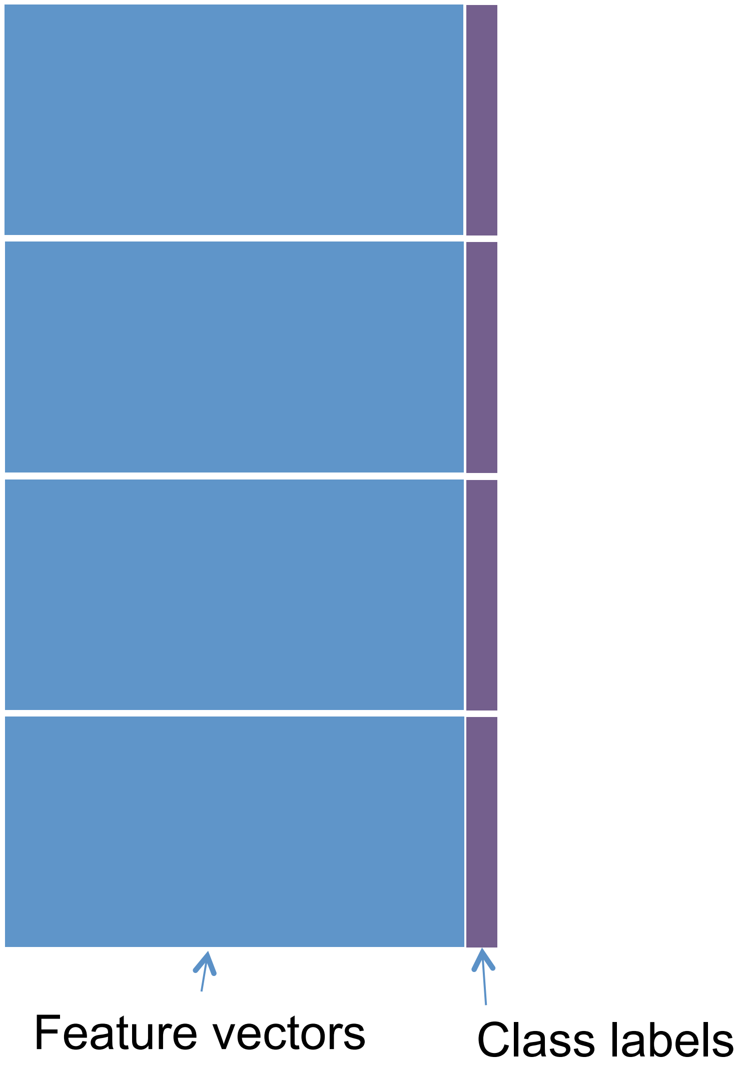


Each row is a set of features and a corresponding class label

$$\left(x_0{}^{(i)}, x_1{}^{(i)}, \cdots, x_{p-1}{}^{(i)}, t^{(i)}\right)$$

Cross Validation

Partition the given set into P subsets

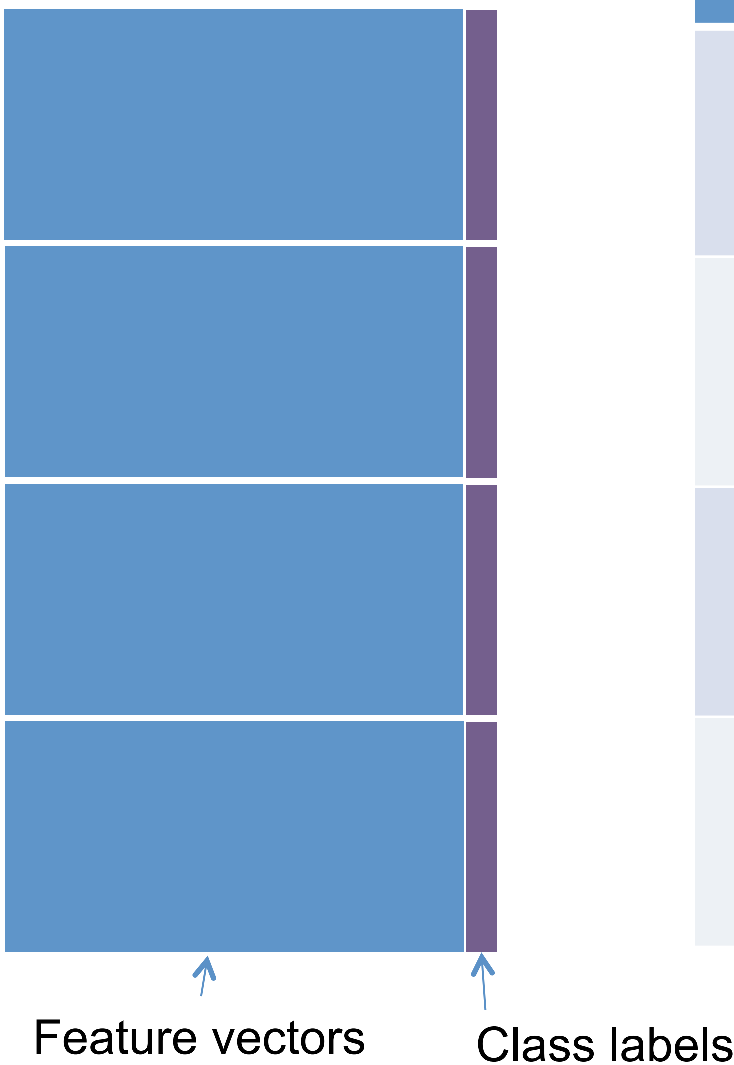


Example:

Let P=4.
Run the train/test steps 4 times.
At each iteration: use one of the partitions as the test set and combine the other 3 as the training set.
The overall performance is the average of the test performance over the 4 iterations.

Cross Validation

Partition the given set into P subsets



Iteration 0	Iteration 1	Iteration 2	Iteration 3
Test	Train	Train	Train
Train	Test	Train	Train
Train	Train	Test	Train
Train	Train	Train	Test

Plug-in Classifier

At each iteration, assemble the training set.

Use the training set to estimate the parameters.

Then plug those parameters in the classifier to obtain the decision rule.

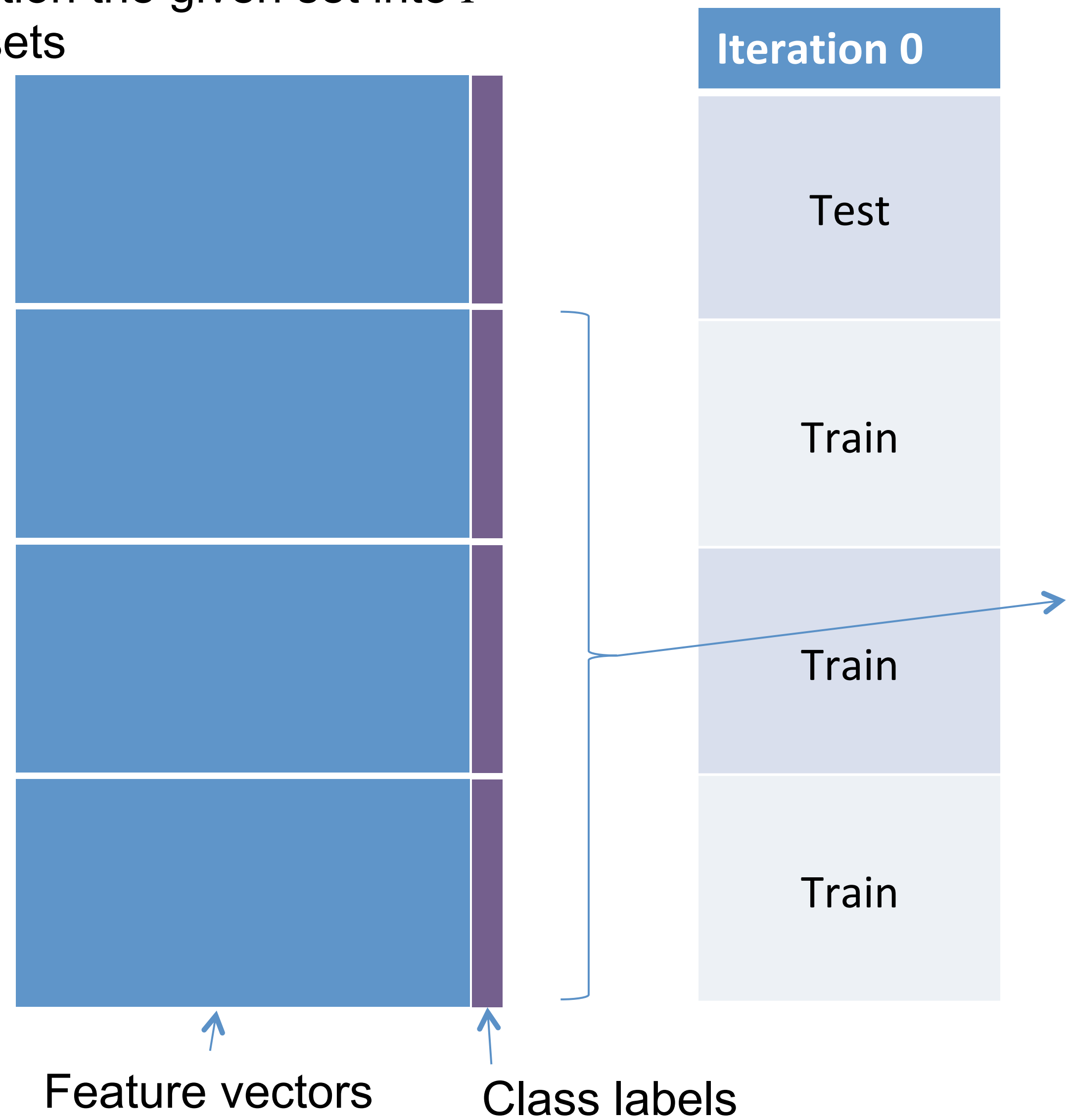
Let $\mathbf{w} = \hat{\mathbf{\Sigma}}^{-T}\left(\hat{\mathbf{m}}_1 - \hat{\mathbf{m}}_0\right)$ and $w_{\mathrm{p}} = -\left(\hat{\mathbf{m}}_1 - \hat{\mathbf{m}}_0\right)^T \hat{\mathbf{\Sigma}}^{-1} \frac{\left(\hat{\mathbf{m}}_1 + \hat{\mathbf{m}}_0\right)}{2} + \log\frac{\hat{\pi}_0}{\hat{\pi}_1}$.

Define $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_p$. The hyperplane $g(\mathbf{x}) = 0$ is the decision boundary.

The optimal decision rule is to choose $l = 0$ if $g(\mathbf{x}) < 0$ and $l = 1$ if $g(\mathbf{x}) > 0$.

Cross Validation

Partition the given set into P subsets



Use these vectors to compute $\hat{\pi}_1, \hat{\pi}_0, \hat{\mathbf{m}}_1, \hat{\mathbf{m}}_0, \hat{\boldsymbol{\Sigma}}$.

Then use the parameters to compute

$$\mathbf{w} = \hat{\boldsymbol{\Sigma}}^{-T}\left(\hat{\mathbf{m}}_1 - \hat{\mathbf{m}}_0\right) \text{and } w_{\mathrm{p}} = -\left(\hat{\mathbf{m}}_1 - \hat{\mathbf{m}}_0\right)^T \hat{\boldsymbol{\Sigma}}^{-1} \frac{\left(\hat{\mathbf{m}}_1 + \hat{\mathbf{m}}_0\right)}{2} + \log\frac{\hat{\pi}_0}{\hat{\pi}_1}.$$

How do we measure performance?

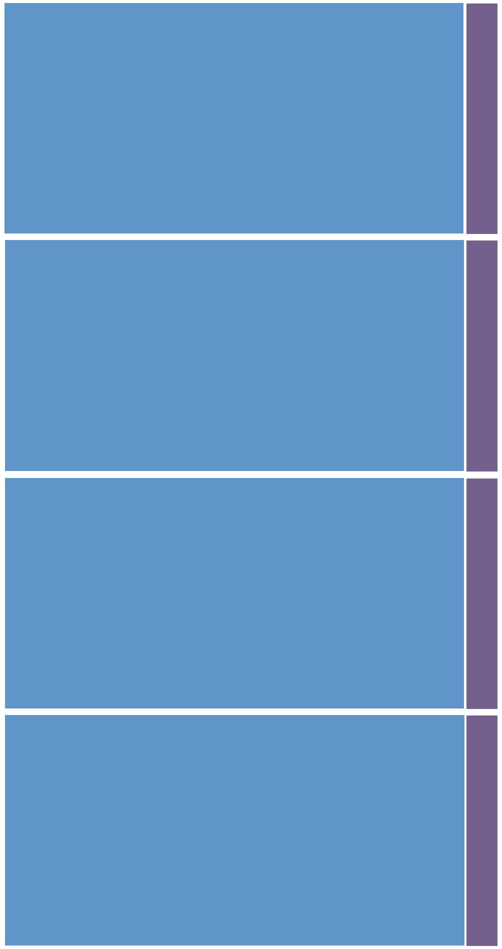
At each cross-validation iteration, use the designated test set to determine the misclassification count and rate

$$
\begin{aligned}
&\text{Use } \mathbf{w} \text{ and } w_{\mathrm{p}} \text{ obtained from the training set.}\\
&\text{for } (i = 0;\ i < \text{number of samples in the test set};\ i++)\ \{\\
&\quad \text{Let the } i\text{th labeled vector in the test set be } \left(\mathbf{x}^{(\mathrm{i})}, t^{(i)}\right).\\
&\quad \text{Compute } g\left(\mathbf{x}^{(i)}\right) = \mathbf{w}^T \mathbf{x}^{(i)} + w_p.\\
&\quad \text{if } \left(\left(g\left(\mathbf{x}^{(i)}\right) < 0\right) \text{and} \left(t^{(i)} == 1\right)\right)\{\ \ \text{// error!}\\
&\qquad \text{misclassification}++;\\
&\quad \}\\
&\quad \text{else if } \left(\left(g\left(\mathbf{x}^{(i)}\right) > 0\right) \text{and} \left(t^{(i)} == 0\right)\right)\{\ \ \text{// error!}\\
&\qquad \text{misclassification}++;\\
&\quad \}\\
&\}
\end{aligned}
$$

Divide misclassification count by the size of the test set to get the misclassification count

Cross Validation

Partition the given set into P subsets



Iteration 0	Iteration 1	Iteration 2	Iteration 3
Test	Train	Train	Train
Train	Test	Train	Train
Train	Train	Test	Train
Train	Train	Train	Test

Average the test performance over the 4 iterations

How do we measure performance?

At each cross-validation iteration, use the designated test set to determine the misclassification count and rate

Use $\mathbf{w}$ and w_{p} obtained from the training set.

for ($i = 0$; $i <$ number of samples in the test set; $i{+}{+}$) {

Let the ith labeled vector in the test set be $\left(\mathbf{x}^{(\mathrm{i})}, t^{(i)}\right)$.

Compute $g\left(\mathbf{x}^{(i)}\right) = \mathbf{w}^T\mathbf{x}^{(i)} + w_p$.

if $\left(\left(g\left(\mathbf{x}^{(i)}\right) < 0\right)$ and $\left(t^{(i)} == 1\right)\right)$ { // error!

misclassification++;

}

else if $\left(\left(g\left(\mathbf{x}^{(i)}\right) > 0\right)$ and $\left(t^{(i)} == 0\right)\right)$ { // error!

misclassification++;

}

}

Two types of errors

How do we measure performance?

At each cross-validation iteration, use the designated test set to determine the misclassification count and rate

$$\text{Use } \mathbf{w} \text{ and } w_{\mathrm{p}} \text{ obtained from the training set.}$$

$$\text{for } (i = 0;\, i < \text{number of samples in the test set};\, i++)\ \{$$

$$\quad \text{Let the } i\text{th labeled vector in the test set be } \left(\mathbf{x}^{(\mathrm{i})}, t^{(i)}\right).$$

$$\quad \text{Compute } g\left(\mathbf{x}^{(i)}\right) = \mathbf{w}^T \mathbf{x}^{(i)} + w_p.$$

$$\quad \text{if } \left(\left(g\left(\mathbf{x}^{(i)}\right) < 0\right) \text{and} \left(t^{(i)} == 1\right)\right)\{\ \ // \text{ error!}$$

$$\qquad \text{misclassification} ++;$$

$$\quad \}$$

$$\quad \text{else if } \left(\left(g\left(\mathbf{x}^{(i)}\right) > 0\right) \text{and} \left(t^{(i)} == 0\right)\right)\{\ \ // \text{ error!}$$

$$\qquad \text{misclassification} ++;$$

$$\quad \}$$

$$\}$$

True class is 1 but the classifier decided 0

How do we measure performance?

At each cross-validation iteration, use the designated test set to determine the misclassification count and rate

Use $\mathbf{w}$ and w_{p} obtained from the training set.

$\text{for } (i = 0; i < \text{number of samples in the test set}; i{+}{+})\ \{$

$\quad$ Let the ith labeled vector in the test set be $\left(\mathbf{x}^{(\mathrm{i})}, t^{(i)}\right)$.

$\quad$ Compute $g\left(\mathbf{x}^{(i)}\right) = \mathbf{w}^T\mathbf{x}^{(i)} + w_p$.

$\quad \text{if } \left(\left(g\left(\mathbf{x}^{(i)}\right) < 0\right) \text{and} \left(t^{(i)} == 1\right)\right)\{\ \text{// error!}$

$\qquad \text{misclassification}{+}{+};$

$\quad \}$

$\quad \text{else if } \left(\left(g\left(\mathbf{x}^{(i)}\right) > 0\right) \text{and} \left(t^{(i)} == 0\right)\right)\{\ \text{// error!}$

True class is 0 but the classifier decided 1

$\qquad \text{misclassification}{+}{+};$

$\quad \}$

$\}$

Types of Errors

0: normal; 1: significant (‘defective’)

- True class is 1 but classifier decided 0
  - Missed detection
- True class is 0 but classifier decided 1
  - False alarm

Both types are errors but the consequences may not be the same

Confusion Matrix

		True Class 0	True Class 1
Classifier Decision	0	True Negatives	False Negative (Missed Detection)
	1	False Positive (False Alarm)	True Positives

Performance Measures

- Besides “misclassification rates”, there are other, often domain-specific, measures:
  - Accuracy
  - Recall, Sensitivity
  - Specificity
  - Fallout
  - Precision

Survey of Performance Measures

	Class: 0	Class: 1	
Decision	TN	FN	0
	FP	TP	1

- Accuracy

  (# True Positives + # True Negatives) / *N*

- Recall, Sensitivity

  (# True Positives) / (True Positives + True Negatives)

- Specificity

  (# True Negatives) / (# True Negative + # False Positives)

- Precision

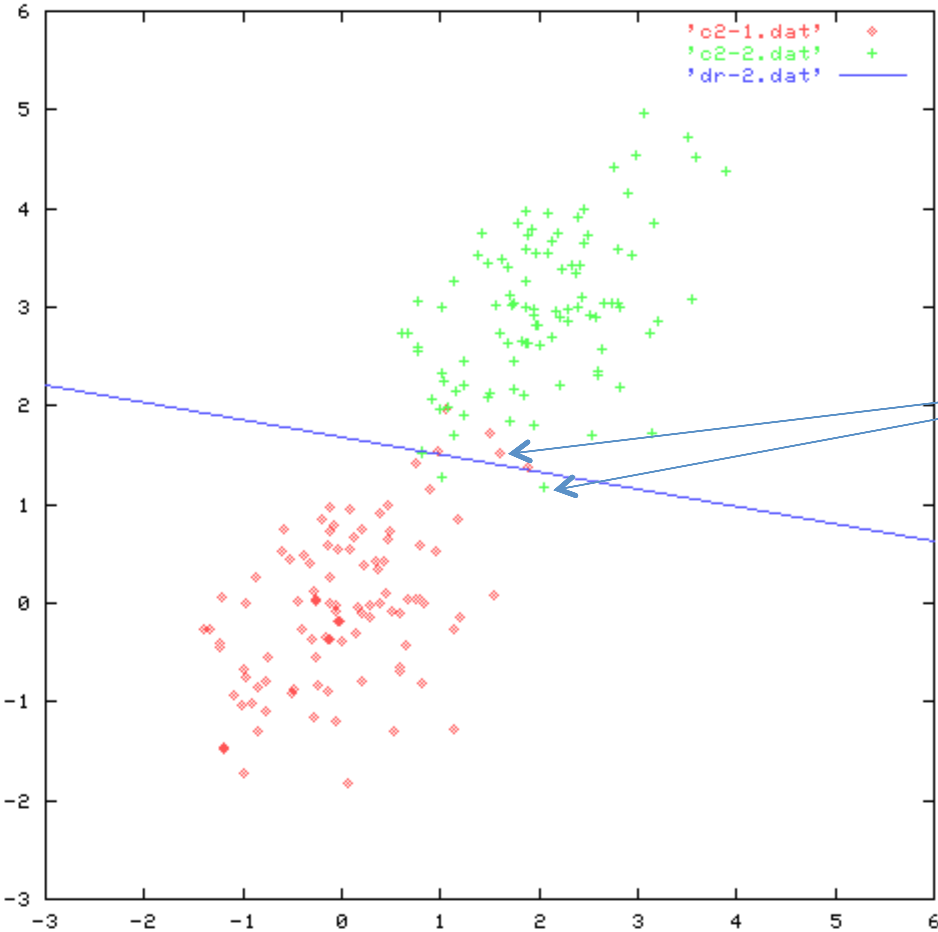
  (# True Positive) / (# True Positive + # False Positive)

- Fallout

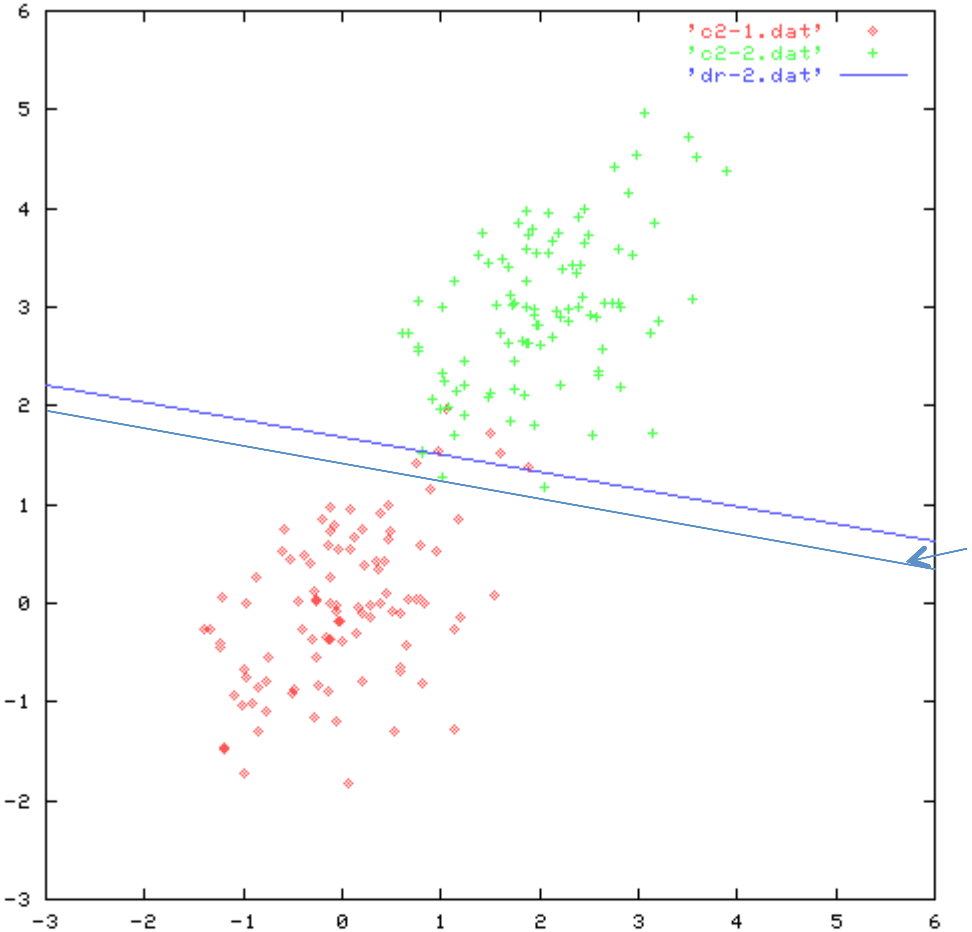
  (# False Positive) / (# True Negative + # False Positive)

Missed Detection vs False Alarm

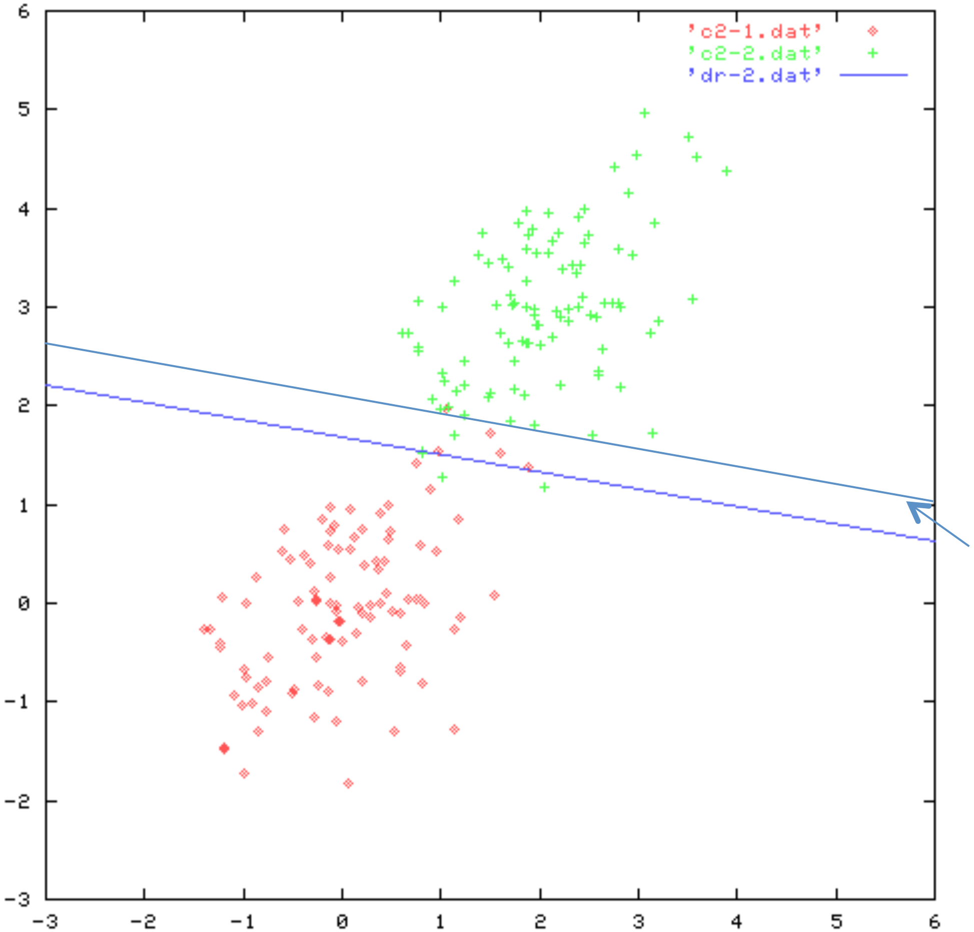
- Usually trade off one type of error for another
- Usually by adjusting the classifier parameters



This classifier has errors of both types



This classifier has no missed detections, but plenty of false alarms



This classifier has no false alarms, but plenty of missed detections