Pattern Recognition & Classification

Classification

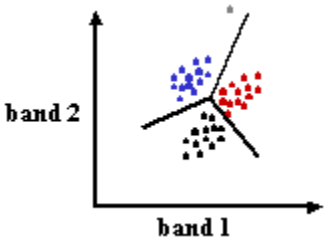
- **Supervised**
  - parallelpiped
  - minimum distance
  - maximum likelihood (Bayes Rule)
    - > non-parametric
    - > parametric
- **Unsupervised (clustering)**
  - K-Means
  - ISODATA

Pattern Recognition

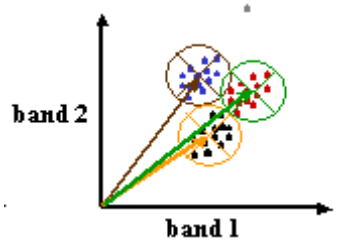
- Pattern recognition in remote sensing has been based on the intuitive notion that ***pixels belonging to the same class should have similar gray values in a given band***.
  - Given two spectral bands, pixels from the same class plotted in a two-dimensional histogram should appear as a localized cluster.
  - If n images, each in a different spectral band, are available, pixels from the same class should form a localized cluster in n-space.

What is a pattern?

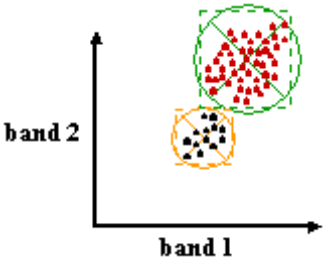
For the purpose of standard classification methods, a ***pattern*** is a cluster of data points in an n-dimensional feature space, and ***classification*** is the procedure for discriminating that cluster from other data sources in the feature space.



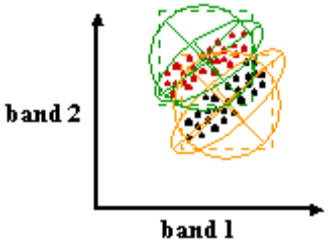
- Focus on distinguishing between pairs of clusters
- clusters separated by lines (or surfaces in n-dimensions)
- 1 line for each pair of clusters



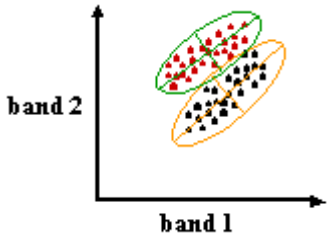
- focus on fully describing each cluster
- each cluster is specified with
  - mean vector
  - distribution (e.g., circle, rectangle, etc.)



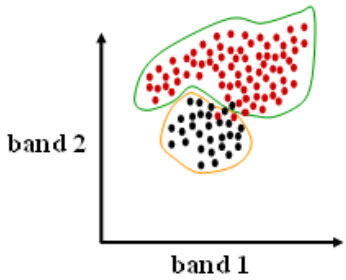
- clusters described by simple distributions
  1. mean vector & circle (variable radius)
  2. rectangle



- simple shapes may not describe the actual geometry of cluster.
  - rectangle, circle, ellipse

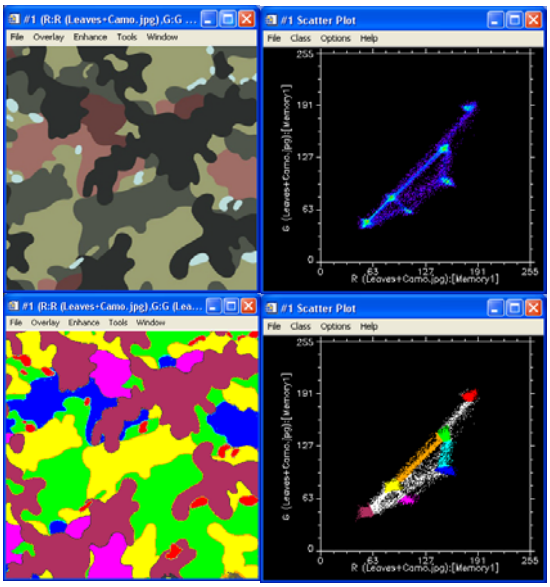


Standard Max-Likelihood: cluster specified with
- mean vector
- ellipse

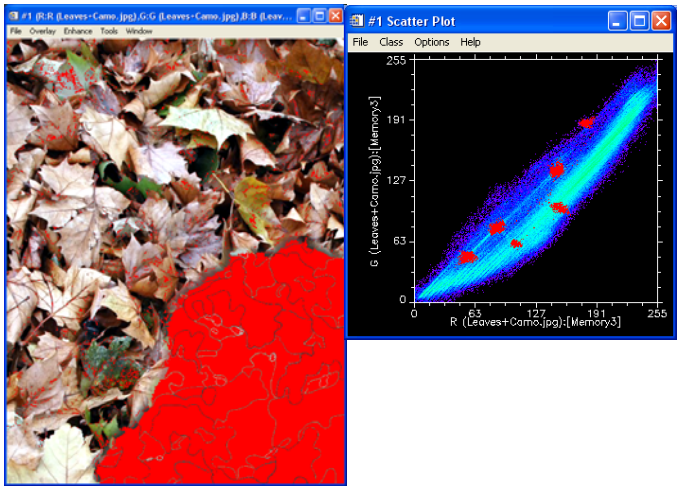


- More general: cluster adapted to sample distribution

- Pattern recognition in remote sensing has been based on the intuitive notion that ***pixels belonging to the same class should have similar gray values in a given band***.
- Variations in a cluster will occur even if the pattern is very well defined (e.g., quantization noise, atmospheric variability, illumination differences, or any number of other 'natural" and instrumental sources of variability (e.g., mixed pixels).
- If several patterns (classes) appear as **distinct** clusters then the classes are discriminable.



Camouflage uses dyes with a very narrow color range, and the colors plot in narrowly defined regions in color space. The connecting lines in the scatterplot are due to mixed pixels on the boundary between two colors.



The distribution of natural materials (e.g., leaves) in color is much broader and may not exhibit distinct cluster

- Real objects (targets) tend to exhibit a much broader distribution in measurement space.
  - Clusters are often less distinct
  - Overlap between clusters in common
  - Some misclassification is generally unavoidable

Classification

• Classification is a procedure for sorting pixels and assigning them to specific categories.

• Characterize pixels using features
- original band gray values
- algebraic combinations of the original bands
- texture measures
- ….

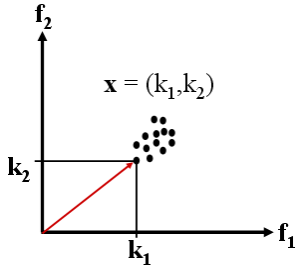
• The set of characterizations is called a **feature vector**

e.g., $\mathbf{x} = (k_1, k_2, k_3)$ where, for example:

k_1 = gray value in band 1
k_2 = gray value in band 2
k_3 = ratio of gray values in bands 4 and 3

Feature Vector

A **feature vector**, **x**, locates a pixel in **feature space**.
Feature space can be thought of as an n-dimensional scatterplot with axes representing the **derived spectral or spatial features**.



Definitions

Feature (Measurement) space:
Measurement space can be thought of as an n-dimensional scatterplot whose axes represent the gray values in the original bands. If some or all of the original bands are replaced with other variables it is called feature space.

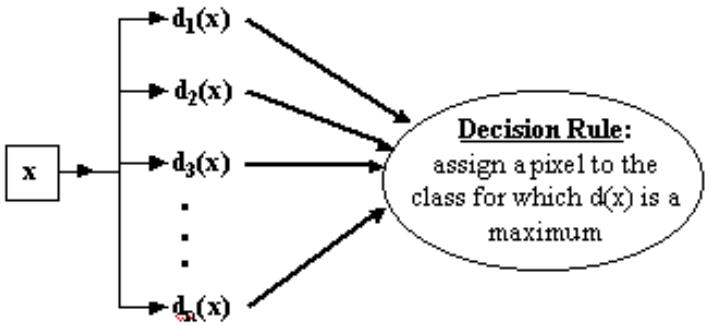
Feature vector (x): $\mathbf{x} = (k_1, k_2, k_3, ..., k_n)$
The feature (or measurement) vector locates a pixel in feature (or measurement) space.

Discriminant function, $d_i(x)$
A function of the measurement vector, **x**, which provides a measure of the probability that **x** belongs to the i^{th} class. A distinct discriminant function is defined for each class, under consideration.

Decision Rule
The rule which assigns pixels to a particular class based on a comparison of the discriminant function for each class.

Classification is a procedure for sorting pixel and assigning them to specific categories.

x = feature (or measurement) vector
$d_i(\mathbf{x})$ = discriminant function



There are two general types of classification techniques: **supervised** and **unsupervised** classification.

A classification procedure is ***supervised*** if the user
- defines the decision rules for each class directly or
- provides training data (class prototypes) for each class to guide the computer classification.

A classification procedure is ***unsupervised*** if
- no training data are required
- the user needs to specify the number of classes (at most)

A. Supervised Classification

A classification procedure is said to be supervised if the user either defines the decision rules for each class directly or provides training data for each class to guide the computer classification.

General procedure

1. ***Set up a classification scheme*** - Determine the classes into which the pixels are to be assigned. The categories need to:
   a. be appropriate to the scale and resolution (both spectral and spatial) of the image data
   b. be appropriate to the application
   c. include background classes that can be confused with the target classes.
2. ***Select the features to be used in the classification*** (Feature Extraction)
   a. eliminate redundant of ineffective bands.
   b. define useful spectral or spatial features.
3. ***Characterize the classes in terms of the selected features***
   a. Provide spectral reflectance measures for each class. These may be laboratory or field spectra.
   b. Select at least two subsets of data from the image to represent each of the desired categories. One subset will be used to "train" the classifier, the other will be used to evaluate (test) the results of the classification. Ideally, these subsets would be selected independently and should not overlap.
4. ***Determine the parameters (if any) required for the classifier*** - Use the training data to estimate the parameters required by the particular classification algorithm to be used. (Define the discriminant functions.)
5. ***Perform the classification*** - Evaluate the discriminant functions and assign pixels to classes by applying the decision rule for each pixel.
6. ***Evaluate the results*** - Use the test data to evaluate the accuracy of the classification.

Supervised classifiers differ primarily in

the definition of the discriminant function.

Parallelepiped Classifier

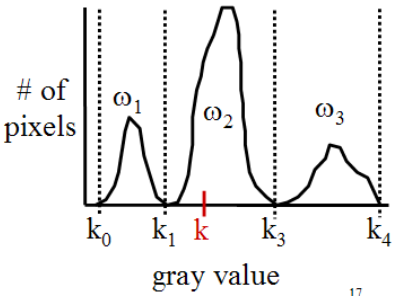
– The parallelepiped classifier is essentially a **thresholding operation** in multiple bands.
– The simplest case is with a single variable (1 spectral band) where a pixel is assigned to a particular class if its gray value is less than some minimum and greater that some maximum, i.e., in pseudocode:

For the i^{th} class,

if $k_{i,min} < k < k_{i,max}$
then $d_i(k) = 1$
else $d_i(k) = 0$
endif

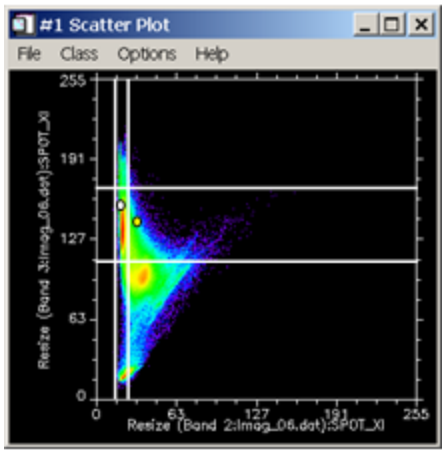
$d_1(k)=0;\ d_2(k)=1;\ d_3(k)=0$

$k \in w_2$



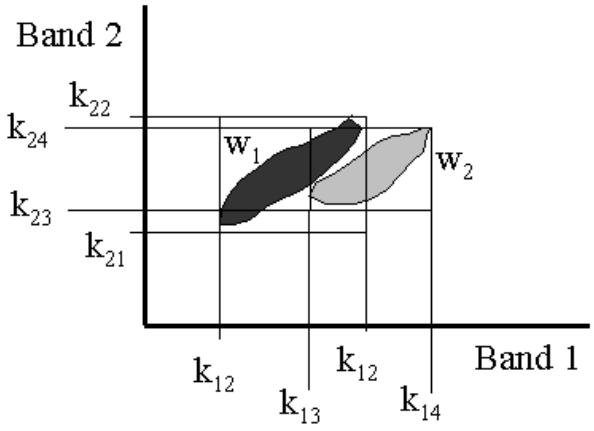
– For the n-band case, and the ith class:

for the k^{th} sample
for i = 1:nclass
$d_i(k) = 0$
for b = 1:nband
if $k_{b,i,min} < k_1 < k_{b,i,max}$
$d_i(k) = d_i(k) + 1$
end if
end
if $d_i(k) <$ nband then $d_i(k) = 0$
end
end



Advantage: The parallelepiped classifier models the data distribution as a rectangle in measurement space. It is simple to set up, easy to understand and very fast (real time).

Disadvantage: Problems arise with the parallelepiped classifier when the actual data distribution does not fit the rectangular model well. For example, consider the two-class case:



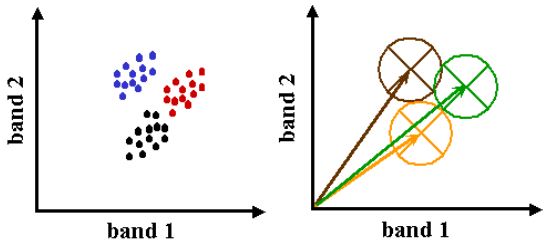
Even though the classes do not overlap, the rectangular boundaries defined by the classifier do overlap.

Within the overlap region the classifier cannot distinguish between the two classes.

A more powerful and adaptable classification scheme is needed.

Minimum Distance Classifier

The minimum distance classifier defines classes in terms of the distance from a prototype vector – usually the mean vector for the class. The discriminate function is defined in terms of distance from the mean: $\mathbf{d_i(k) = 1/(\mu_i - k)}$ where μ_i is the mean vector for the i^{th} class.

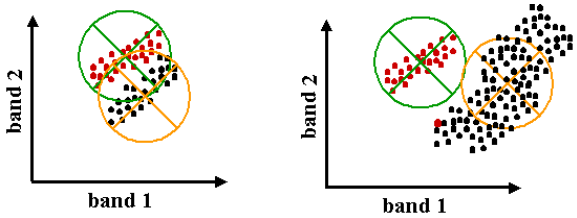


Advantage

- somewhat better description of the data distribution.

Disadvantage

- much more computation than for the parallelepiped method. (2*m*n where n = # bands and m= # pixels)



Maximum-likelihood classification

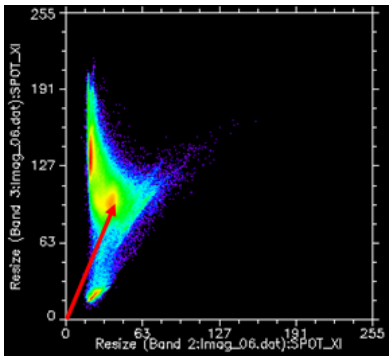
Maximum-likelihood classification is a general term which encompasses the most common supervised classifiers. All maximum-likelihood classifiers derive ultimately from Baye's Theorem:

$$p(\omega_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_i)\, p(\omega_i)}{p(\mathbf{x})} \qquad \textbf{Baye's Theorem} \qquad (7.1)$$

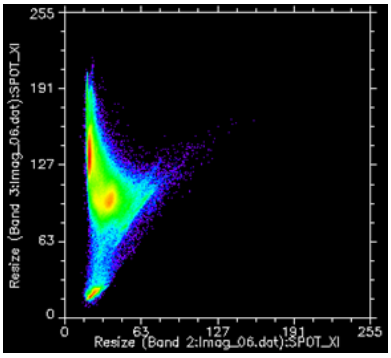
where:

$\mathbf{x}$ = measurement vector
ω_i = the ith class
$p(\mathbf{x})$ = the probability density function (normalized histogram)
$p(\mathbf{x}|\omega_i)$ = the class-conditional probability density function (normalized class histogram)
$p(\omega_i)$ = the prior probability
$p(\omega_i|\mathbf{x})$ = the posterior probability

$\mathbf{x}$ A **measurement vector** is simply an ordered list of the value (grey value, digital number) of a pixel in each of the image channels. That is, $\mathbf{x} = (x_1, x_2, x_3, \ldots, x_n)$ where x_i is the grey value of the pixel in the i^{th} band.

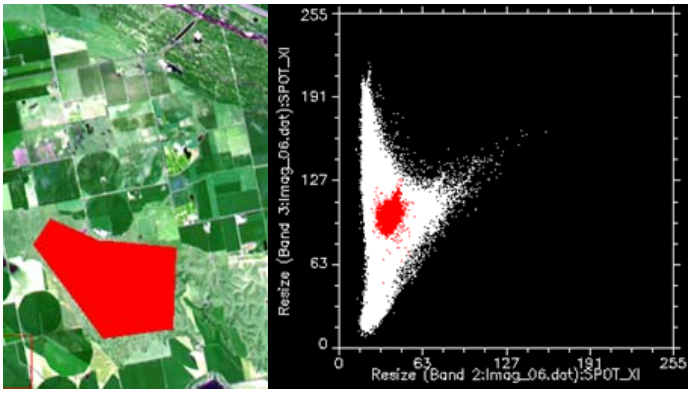


p(**x**) The ***probability density function*** (normalized histogram)



- The probability that a pixel ***selected at random*** from the image will have the measurement vector **x**.
- This is the ***"joint probability"*** that the pixel will have a value of x_1 in band 1, x_1 in band 2, etc.
- Approximated by the ***n-dimensional histogram*** of the image, divided by the total number of pixels in the image.

$p(\mathbf{x}|\omega_i)$ The ***class-conditional probability density function*** (normalized class histogram)



- The probability that a pixel chosen randomly from all members of class ω_i will have the measurement vector x.
- $(\mathbf{x}|\omega_i)$ is usually estimated from training data, in which case p(x|w i) is the number of times that a pixel with characteristics, x, occurs in the training data, divided by the total number of pixels in the training set.

$p(\omega_i)$ The ***prior probability*** (or *a priori* probability)

- the probability that class ω_i will occur in the image.
- This is usually determined (if at all) from prior knowledge: preliminary field tests; maps; historical data; experience of the user.
- When there is insufficient information to make any reasonable estimate of, it is common practice to assume that all the probabilities are equal. Classifications based on this assumption tend to favor the most rarely occurring classes.

$p(\omega_i|\mathbf{x})$ The ***posterior probability***

- The probability that a pixel belongs to class ω_i given that it has the characteristics, **x**.
- This is the discriminant function.

Maximum-Likelihood decision rule

Given Baye's Theorem, it is easy to define a rule by which pixels can be sorted among various classes: a pixel will be assigned to the class with the highest posterior probability given that it has the characteristics of the measurement vector **x**. This is the Baye's optimum, or maximum likelihood decision rule:

$$x \in \omega_i \quad \text{iff} \quad \frac{p(\mathbf{x} \mid \omega_i)p(\omega_i)}{p(\mathbf{x})} \geq \frac{p(\mathbf{x} \mid \omega_j)p(\omega_j)}{p(\mathbf{x})} \quad \text{for all } j \neq i$$

Maximum-Likelihood Decision Rule

In practice one defines a **discriminant function**, $d_i(\mathbf{x})$ for each class such that:

$$d_i(x) = \frac{p(\mathbf{x}|\omega_i)p(\omega_i)}{p(\mathbf{x})} \tag{7.2}$$

Each pixel is selected one at a time from the image, the discriminant functions for every class are computed for that pixel, and the pixel is assigned to the discriminant function with the highest value, i.e.,

$$\mathbf{x} \in \omega_i \quad \text{iff} \quad d_i(\mathbf{x}) \geq d_j(\mathbf{x}) \qquad \text{for all } j \neq i$$

The discriminant function is usually simplified as much as possible in order to minimize computation time. The first simplification is to drop the $p(\mathbf{x})$ term. Since $p(\mathbf{x})$ is the same for all classes and since only the relative values of $d_i(\mathbf{x})$ are of concern, classification results will not be altered. In such cases,

$$d_i(\mathbf{x}) = p(\mathbf{x}|\omega_i)\, p(\omega_i) \tag{7.3}$$

When no estimates of $p(\omega_i)$ are possible, the typical assumption is that $p(\omega) = 1/n_c$ where n_c is the number of classes. The expression for $d_i(\mathbf{x})$ can be simplified even further to:

$$d_i(\mathbf{x}) = p(\mathbf{x}|\omega_i) \tag{7.4}$$

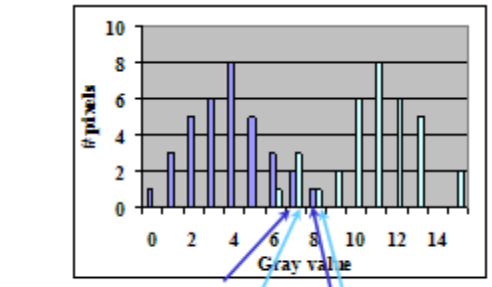
Non-parametric classifiers

If the class-conditional probability density function, $p(\mathbf{x}|\omega_i)$, is estimated by using the frequency of occurrence of the measurement vectors in the **training data**, the resulting classifier is non-parametric. An important advantage of the non-parametric classifiers is that any pattern, however irregular it may be, can be characterized exactly. This advantage is generally outweighed by two difficulties with the non-parametric approach:

1) It is difficult to obtain a large enough training sample to adequately characterize the probability distribution of a multi-band data set.
2) Specification of a meaningful n-dimensional probability density function requires a massive amount of memory or very clever programming, and

Example: Consider the 1-band, two-class problem for which training data are given in the table below. Both training sets have 34 samples.

	class 1		class 2	
k	**n**	**p(k|w)**	**n**	**p(k|w)**
0	1	0.029	0	0.000
1	3	0.088	0	0.000
2	5	0.147	0	0.000
3	6	0.176	0	0.000
4	8	0.235	0	0.000
5	5	0.147	0	0.000
6	3	0.088	1	0.029
7	2	0.059	3	0.088
8	1	0.029	1	0.029
9	0	0	2	0.059
10	0	0	6	0.176
11	0	0	8	0.235
12	0	0	6	0.176
13	0	0	5	0.147
14	0	0	0	0.000
15	0	0	2	0.059
sum	34	1	34	1



$$d_i(\mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_1)\, P(\omega_1)}{p(\mathbf{x})}$$

$p(7) = 5/68 = 0.0735$ $\quad$ $p(8) = 2/68 = 0.0294$

$p(7|\omega_1) = 2/34 = 0.059$ $\quad$ $p(8|\omega_1) = 1/34 = 0.029$

$p(7|\omega_2) = 3/34 = 0.088$ $\quad$ $p(8|\omega_2) = 1/34 = 0.029$

If we assume that both classes have an equal prior probability: $P(\omega_1) = P(\omega_2) = 0.5$

$$d_1(7) = \frac{0.059 * 0.5}{0.0735} = 0.4 \qquad d_2(7) = \frac{0.088 * 0.5}{0.0735} = 0.6 \quad \rightarrow \quad 7 \in \omega_2$$

$$d_1(8) = \frac{0.029 * 0.5}{0.0294} = 0.5 \qquad d_2(8) = \frac{0.0294 * 0.5}{0.0294} = 0.5 \quad \rightarrow \quad \text{tie}$$

More generally, the discriminant function for class 1 is then given by: $p(x|\omega_1)\, p(\omega_1)$. Any pixel for which $p(x|\omega_1)\, p(\omega_1) > p(x|\omega_2)\, p(\omega_2)$ will be assigned to class 1 and vice versa. Letting $p(\omega_1) = 0.7$ and $p(\omega_2) = 0.3$, the class assignments change:

$d_1(8) = p(8|\omega_1)\, p(\omega_1) = (1/34) * 0.7 = 0.021$ ➔ $8 \in \omega_1$

$d_2(8) = p(8|\omega_2)\, p(\omega_2) = (1/34) * 0.3 = 0.009$

and

$d_1(7) = p(7|\omega_1)\, p(\omega_1) = (2/34) * 0.7 = 0.041$ ➔ $7 \in \omega_1$

$d_2(7) = p(7|\omega_2)\, p(\omega_2) = (2/34) * 0.3 = 0.018$

The problem of sparse training data is more serious than the computing limits. If 10 pixels were marginally sufficient to define the gray value distribution in the one band example above, then one might expect to need 10^2=100 samples for a training set to define a two-band distribution, and 10^n samples to define an n-band distribution. Thus, for a four-band data set, there should be at least 10^4 pixels for training in each class in order to minimally describe the multispectral data distribution. This is impractical at best and is generally impossible.

A further problem exists. There are 256^n possible values for the class-conditional probability, $p(\mathbf{x}|\omega_i)$, for an n-band, byte data set. It is difficult to define an array for large values of n on most computers.

Although algorithms do exist for implementing the non-parametric maximum-likelihood classifier, addressing both the problems of sparse data and minimizing the data storage requirements, these algorithms have not seen extensive use in remote sensing applications and will not be treated here. The approach taken in remote sensing is to make some assumption about the data distribution and parameterize $p(x|\omega_i)$.

Parametric maximum-likelihood classification

One way to avoid the difficulties encountered with a non-parametric classification scheme is to parameterize the data, i.e., to assume that each probability function can be adequately approximated with a mathematically simple functional form. In effect, we try to model the data distribution in feature space in a way that can be easily parameterized. One can visualize the "model" in 2-dimensions as a rectangle, circle, or ellipse that surrounds the data and whose center is located at the data mean. The training data are then used to find parameters that will optimize the fit of the assumed function to the data distribution. This will require far fewer samples for the training data set.

A ***parametric maximum-likelihood classifier*** uses a parameterized model to describe the distribution $p(\mathbf{x}|\omega_i)$ Since the functional form of a normal distribution is well defined, the probabilities are completely specified when the means, variances, and covariances are specified.

Assumption: We assume that the data can be modeled adequately by a
multi-normal distribution

Implications

- *For an n-band classification, we will need at least 10*n pixels*

  Swain, P.H, and S.M. Davis, 1978. *Remote Sensing: The Quantitative Approach* (New York, McGraw-Hill)

- *Classification accuracy will **decrease** if there are less than 10*n pixels for the training of that class.*

  Derde, M. P. and D.L. Massart, 1989. Evaluation of the required sample size in some supervised pattern recognition techniques," *Analytica Chimica Acta*, vol. 223(1):19–44, 1989

  F. Tsai & W. Philpot, 2002. A Derivative-Aided Hyperspectral Image Analysis System for Land-Cover Classification. *Transactions on Geoscience and Remote Sensing*, 40(2):416-425

The single-band case

In the 1-dimensional (one-band) case, the data distribution for the i^{th} class is completely specified by a mean, m_i, and a variance, σ_i^2 (or standard deviation, σ_i). Specifically, the class-conditional probability is given by:

$$p(\mathbf{x}|\omega_i) = \frac{1}{\sigma_i (2\pi)^{1/2}} \exp\left[-\frac{(\mathbf{x}-\mu_i)^2}{2\sigma_i^2} \right]$$

and the discriminant function is given by:

$$d_i(\mathbf{x}) = p(\mathbf{x}|\omega_i) p(\omega_i)$$
$$= \frac{1}{\sigma_i (2\pi)^{1/2}} \exp\left[-\frac{(\mathbf{x}-\mu_i)^2}{2\sigma_i^2} \right] p(\omega_i)$$

Computations may be further simplified by

1) eliminating any terms that do not affect the results and

2) by performing certain operations ahead of time and storing the results.

To this end, it is common practice to use a log transformation and to ***redefine*** the discriminant function as:

$$d_i(\mathbf{x}) = -\ln(\sigma_i) - \frac{1}{2}\ln(2\pi) - \frac{(\mathbf{x}-\mu_i)^2}{2\sigma_i^2} + \ln\left[p(\omega_i)\right]$$

since this eliminates the need for an exponential function and does not change the relative values of the $d_i(x)$. The first term may also be eliminated since it is independent of class and does not alter the comparison of classes. Thus, the discriminant function may be defined as:

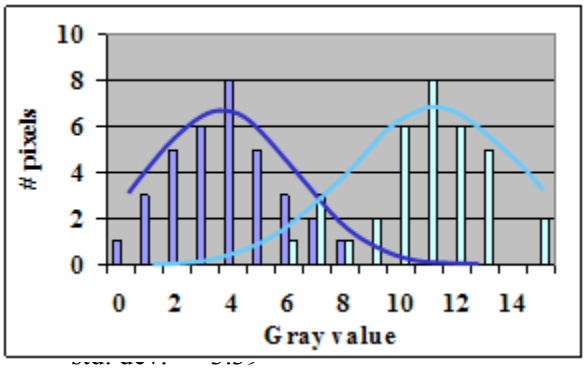
$$d_i(\mathbf{x}) = -\ln(\sigma_i) - \frac{(\mathbf{x}-\mu_i)^2}{2\sigma_i^2} + \ln\left[p(\omega_i)\right]$$

Consider the simple example of two classes of equal probability, ω_1 and ω_2, using the same training set data as in the non-parametric example. Histograms for the training sets of the two classes are shown in the figure below. Again, each training set has N samples (N=34). The normal distribution can then be used to estimate the probability function.

Consider the 1-band example:

We need only enough pixels to specify a mean and variance for each class.

Class 1		Class 2	
mean	3.53	mean	10.56
variance	3.58	variance	4.63
std. dev.	1.89	std. dev.	2.15



In this case, regardless of the actual number of pixels in the training set for any gray value, the decision boundary occurs at 6.82. Thus, if k < 6.82 the pixel belongs to ω_1 and if k > 6.82 the pixel belongs to ω_2. Note that, if k is an integer, there is no chance that there will be any ambiguity in the classification.

For pixels with a gray value of 7 the discriminant functions yield:

$$d_1(7) = -\ln(1.89) - (7 - 3.53)^2 / (2*3.58) + \ln p(\omega_1)$$

$$d_2(7) = -\ln(2.15) - (7 - 10.56)^2 / (2*4.63) + \ln p(\omega_2)$$

$$d_1(8) = -\ln(1.89) - (8 - 3.53)^2 / (2*3.58) + \ln p(\omega_1)$$

$$d_2(8) = -\ln(2.15) - (8 - 10.56)^2 / (2*4.63) + \ln p(\omega_2)$$

If we assume that the prior probabilities are equal, i.e., $p(\omega_1) = p(\omega_2) = 0.5$, then

$$d_1(7) = -0.638 - 1.682 + \ln p(0.5) = -6.788$$

$$d_2(7) = -0.766 - 1.369 + \ln p(0.5) = -5.365$$

$$d_1(8) = -0.638 - 2.791 + \ln p(\omega_1) = -8.883$$

$$d_2(8) = -0.766 - 0.708 + \ln p(\omega_2) = -6.985$$

- Since $d_2(7) > d_1(7)$ – i.e., it is less negative – pixels with a gray value of 7 or less will be assigned to ω_2 and since $d_2(8) > d_1(8)$, pixels with gray values of 8 or more will be assigned to class ω_2.
- Since the distributions are continuous, it is even possible to define a ***decision boundary*** at which $d_1(x) = d_2(x)$. In this case $d_1(x) = d_2(x)$ when x = 6.82.
- Note that it is now necessary to know only the mean and standard deviation of the class in order to compute the probabilities for any gray value. By assuming a particular form for the data distribution, computation has been substituted for memory capacity.
- It is also possible to use ***sparse training data*** to characterize the class, and probability ties are rare.

The n-band case.

In the n-band case, the expression for the discriminant function becomes:

$$d_i(\mathbf{x}) = -\frac{1}{2\pi|\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t \Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right] * \ln\left[p(\omega_i)\right]$$

where the **variance, σ_I** is replaces but the **covariance matrix, Σ_i**. As with the one-band case taking the natural log of the above equation yields an equivalent expression for the discriminant function:

$$d_i(\mathbf{x}) = -\frac{1}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma_i| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t \Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) + \ln\left[p(\omega_i)\right]$$

Again, since the first term is independent of class, it will not contribute to the class comparison and we may simplify the discriminant function to:

$$d_i(\mathbf{x}) = -\frac{1}{2}\ln|\Sigma_i| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t \Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) + \ln[p(\omega_i)]$$

– The first and last terms in the above expression need only be computed once since neither varies with gray value. Both depend only on class.

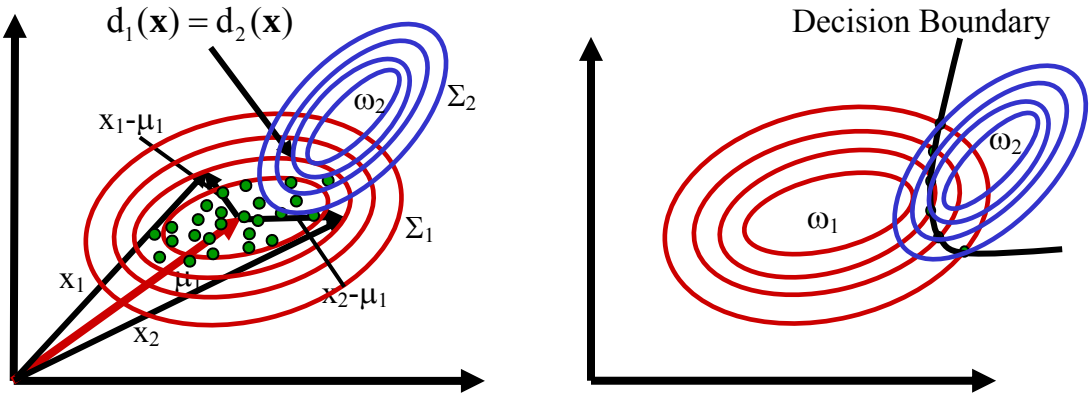
– The middle term describes an n-dimensional ellipsoidal distribution centered on the mean vector, $\boldsymbol{\mu}_i$.

– The matrix product: $(\mathbf{x}-\boldsymbol{\mu}_i)^t \Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)$ is called the ***Mahalanobis Distance***.

The decision boundary between any two classes is a surface in the n-dimensional space defined by the set of points for which the discriminant functions of the two classes are equal, i.e., where:

$$d_1(\mathbf{x}) = d_2(\mathbf{x})$$

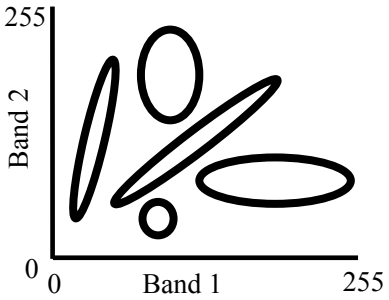
In the illustration below, all points on a single ellipse have equal probabilities of belonging to the class whose mean is located at the center of the ellipse. Points at the intersection of two ellipses of equal probability are on the decision boundary. The decision boundary connects all the points of equal probability for the two classes.



The multinormal discriminant function is the most common form of the maximum-likelihood classifier. Unless otherwise stated, it is this form that is implied when one refers to maximum-likelihood classification.

- The multinormal distribution models a data distribution as an n-dimensional ellipsoid in feature space.
- The size, orientation and ellipticity of the ellipsoid are completely variable.

There are no simplifications to the formula that can be made without significantly altering the nature of the model.



Simplification 1: $\Sigma_i = c_i \Sigma$

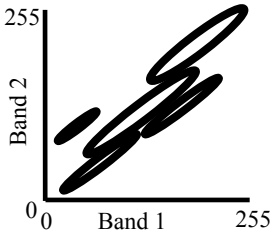
If the classes can all be assumed to vary in a similar fashion being differentiated only by the **mean vector** and the **magnitude of the variance** then some computational efficiency can be gained.

The discriminant function can be simplified to:

$$d_i(\mathbf{x}) = -\frac{1}{2}\ln(c_i \Sigma) - \frac{c_i}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) + \ln\left[p(\omega_i)\right] \qquad (7.13)$$

The major simplification in this form of the discriminant function is that the Mahalanobis distance has been replaced by a simpler function.

$$\Sigma_i = c_i \Sigma = c_i \begin{pmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} & \cdots \\ \sigma_{21} & \sigma_{22} & \sigma_{23} & \\ \sigma_{31} & \sigma_{32} & \sigma_{33} & \\ \vdots & & & \ddots \end{pmatrix}$$

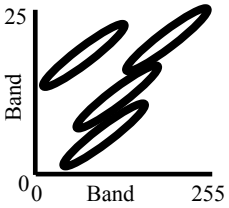


Simplification 2: $\Sigma_i = \Sigma$

In this variation, the covariance matrix is assumed to be identical (in size, ellipticity and orientation) for all classes. Only the location (**mean vector**) of the ellipse actually varies. In this case:

$$d_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\mu_i)^t \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mu_i) + \ln\left[p(\omega_i)\right] \qquad (7.13)$$

$$\Sigma_i = \Sigma = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} & \cdots \\ \sigma_{21} & \sigma_{22} & \sigma_{23} & \\ \sigma_{31} & \sigma_{32} & \sigma_{33} & \\ \vdots & & & \ddots \end{pmatrix}$$

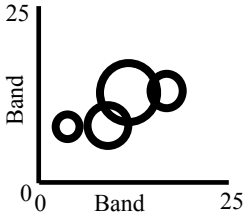


Simplification 3: $\Sigma_i = \sigma_i^2 \mathbf{I}$; $\mathbf{I}$ = the identity matrix

In this case all classes are represented by **spherical** distributions of different sizes. Only the **mean vector** and the **magnitude of the variance** of each class varies. The discriminant function may be reduced to:

$$d_i(\mathbf{x}) = -\frac{1}{2}\ln(\sigma_i) - \frac{1}{2}\|\mathbf{x}-\boldsymbol{\mu}_i\|^2 + \ln\left[p(\omega_i)\right] \qquad (7.14)$$

$$\Sigma_i = \sigma_i \mathbf{I} = \sigma_i \begin{pmatrix} 1 & 0 & 0 & \cdots \\ 0 & 1 & 0 & \\ 0 & 0 & 1 & \\ \vdots & & & \ddots \end{pmatrix}$$



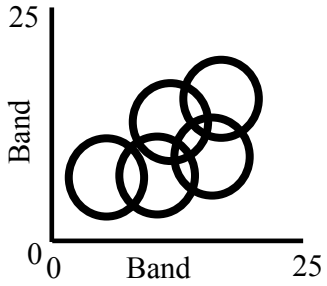
In this case there are immediate benefits in computational simplicity. The matrix operations of the Mahalanobis distance have been replaced by a simple vector product.

Simplification 4: $\Sigma_i = \sigma^2 \mathbf{I}$; $\mathbf{I}$ = the identity matrix

In this case, all class dependence has been removed from the covariance matrix. The result is only a slightly simplified computational scheme, but a severely restricted model. Now, the class data are modeled by a **sphere** and the size of the sphere (variance) is fixed. Only the **mean vector** differentiates among classes. This is the **minimum distance to mean** classifier since a pixel will be assigned to a class if its measurement vector is closer to that class mean than to any other. The simplified discriminant function may be written:

$$d_i(\mathbf{x}) = -\frac{1}{2\sigma}\|\mathbf{x}-\boldsymbol{\mu}_i\|^2 + \ln\left[p(\omega_i)\right]$$

$$\Sigma_i = \sigma I = \sigma \begin{pmatrix} 1 & 0 & 0 & \cdots \\ 0 & 1 & 0 & \\ 0 & 0 & 1 & \\ \vdots & & & \ddots \end{pmatrix}$$



Supervised Classification: general procedure

1. Set up a classification scheme

What are the categories?

Are they appropriate to the application?

Are they appropriate to the scale and resolution of the image data?

2. Define a discriminant function for each class

a. Choose training and test data

Select at least two collections of pixels to represent each class.

- a "training" set to define the discriminant function
- a "test" set to evaluate the results

Ideally these subsets should be selected independently and should not overlap.

b. Determine any parameters required for the classifier

Compute the mean, covariance matrix, etc.

3. Perform the classification

Evaluate the discriminant functions and assign pixels to classes by applying the decision rule for each pixel.

4. Evaluate the results

Use the ***test data*** to evaluate the accuracy of the classification.

Supervised classifiers differ primarily in the definition of the discriminant functions

Error Analysis

Reference Data (test data)

	1	2	3	4	5	6	7	Total
0	x_{01}	x_{02}	x_{03}	$x_{)4}$	x_{05}	x_{06}	x_{07}	Σx_{1j}
1	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	x_{16}	x_{17}	Σx_{1j}
2	x_{21}	x_{22}	x_{23}	x_{24}	x_{25}	x_{26}	x_{27}	Σx_{2j}
3	x_{31}	x_{32}	x_{33}	x_{34}	x_{35}	x_{36}	x_{37}	Σx_{3j}
4	x_{41}	x_{42}	x_{43}	x_{44}	x_{45}	x_{46}	x_{47}	Σx_{4j}
5	x_{51}	x_{52}	x_{53}	x_{54}	x_{55}	x_{56}	x_{57}	Σx_{5j}
6	x_{61}	x_{62}	x_{63}	x_{64}	x_{65}	x_{66}	x_{67}	Σx_{6j}
7	x_{71}	x_{72}	x_{73}	x_{74}	x_{75}	x_{76}	x_{77}	Σx_{7j}
Total	Σx_{i1}	Σx_{i2}	Σx_{i3}	Σx_{i4}	Σx_{i5}	Σx_{i6}	Σx_{i7}	**N**

x_{ij} = # pixels assigned to class i which actually belong to class j
N = total # pixels
N_c = number of classes
Class 0 is for pixels that were not assigned to any defined class.

Example:

Reference (Test) Data

Classified data		water 1	urban 2	HDres 3	LDres 4	forest 5	grass 6	field1 7	field2 8	Total
unclass.	**0**	147	0	3	5	10	15	1	35	216
water	**1**	652	0	0	0	0	0	0	0	652
urban	**2**	1	231	61	11	1	2	0	0	307
HDres	**3**	0	44	465	247	0	27	0	0	783
LDres	**4**	0	0	8	587	205	15	16	0	831
forest	**5**	0	0	0	21	703	4	0	51	779
grass	**6**	0	0	4	3	1	345	0	1	354
field1	**7**	0	7	59	259	0	88	111	0	524
field2	**8**	0	0	0	3	8	45	2	357	415
	Tota	800	282	600	1136	928	541	130	444	4861
									p0 =	3451

Normalized Contingency Matrix

			Reference (Test) Data							
		1	2	3	4	5	6	7	Total
	0	p_{10}	p_{20}	p_{30}	p_{40}	p_{50}	p_{60}	p_{70}	r_0
Classified Data	1	p_{11}	p_{21}	p_{31}	p_{41}	p_{51}	p_{61}	p_{71}	r_1
	2	p_{12}	p_{22}	p_{32}	p_{42}	p_{52}	p_{62}	p_{72}	r_2
	3	p_{13}	p_{23}	p_{33}	p_{43}	p_{53}	p_{63}	p_{73}	r_3
	4	p_{14}	p_{24}	p_{34}	p_{44}	p_{54}	p_{64}	p_{74}	r_4
	5	p_{15}	p_{25}	p_{35}	p_{45}	p_{55}	p_{65}	p_{75}	r_5
	6	p_{16}	p_{26}	p_{36}	p_{46}	p_{56}	p_{66}	p_{76}	r_6
	7	p_{17}	p_{27}	p_{37}	p_{47}	p_{57}	p_{67}	p_{77}	r_7
	Total	c_1	c_2	c_3	c_4	c_5	c_6	c_7	N

x_{ij} = # pixels from reference class i which were classified as class j

N = total # pixels

$p_{ij} = x_{ij} / N$

N_c = number of classes

Example:

		Reference (Test) Data								
		water	urban	HDres	LDres	forest	grass	field1	field2	
		1	2	3	4	5	6	7	8	Total
unclass.	0	0.030	0	0.001	0.001	0.002	0.003	0	0.007	0.044
water	1	0.134	0	0	0	0	0	0	0	0.134
urban	2	0	0.048	0.013	0.002	0	0	0	0	0.063
HDres	3	0	0.009	0.096	0.051	0	0.006	0	0	0.161
LDres	4	0	0	0.002	0.121	0.042	0.003	0.003	0	0.171
forest	5	0	0	0	0.004	0.145	0.001	0	0.010	0.16
grass	6	0	0	0.001	0.001	0	0.071	0	0	0.073
field1	7	0	0.001	0.012	0.053	0	0.018	0.023	0	0.108
field2	8	0	0	0	0.001	0.002	0.009	0	0.073	0.085
	Total	0.165	0.058	0.123	0.234	0.191	0.111	0.027	0.091	1.0

Parameters used in computing error metrics:

total # of pixels = $N = \sum_i \sum_j x_{ij}$ number of classes = N_c

Row sum: unclassified pixels confined to class j=0: $r_j = \sum_{i=1}^{N_c} p_{ij} = p_{1j} + p_{2j} + p_{3j} + \cdots + p_{N_c j}$

Column sum: summation includes unclassified pixels: $c_i = \sum_{j=0}^{N_c} p_{ij} = p_{i0} + p_{i1} + p_{i2} + \cdots + p_{iN_c}$

Diagonal sum: $p_0 = \sum_{j=0}^{N_C} p_{ii} = p_{11} + p_{22} + p_{33} + \cdots + p_{N_C N_C}$

Pixels correctly classified by chance: $p_c = \frac{1}{N^2} \sum_{k=1}^{N_c} \left[\sum_{i=1}^{N_c} x_{ik} \sum_{j=1}^{N_c} x_{kj} \right] = \sum_{k=1}^{N_c} r_k c_k$

Coefficients of agreement:

1. Overall Accuracy (OA)

- The OA includes overall errors of omission without regard to class membership. It disregards errors due to commission entirely and as such represents an overly optimistic estimate of classification accuracy. It might be better named an Overall Producer's accuracy.

$$OA = \frac{\text{Total \# of test pixels correctly classified}}{\text{Total \# of pixels in test sets}}$$

- Overall accuracy:

$$OA = \frac{\sum_i x_{ii}}{\sum_i \sum_j x_{ij}} = \frac{1}{N} \sum_{i=1}^{N_c} x_{ii} = p_0$$

2. Producer's Accuracy (PA)

- Neglects errors of commission but accounts for errors of omission.

$$PA_i = \frac{\text{\# of test pixels correctly classified in class i}}{\text{\# of pixels in test class i}}$$

- Producer's accuracy:

a) **for class j**

$$PA_i = \frac{x_{ii}}{\sum_{j=0}^{N_c} x_{ij}} = \frac{p_{ii}}{c_i}$$

b) **for combined classes**

$$PA_{tot} = \frac{1}{N_c} \sum_{i=1}^{N_c} PA_i = \frac{1}{N_c} \sum_{i=1}^{N_c} \frac{p_{ii}}{c_i}$$

3. User's Accuracy (UA)

- Neglects errors of omission but accounts for errors of commission.

$$UA_i = \frac{\text{\# of pixels correctly classified in class i}}{\text{\# of pixels classified as class i}}$$

- User's accuracy:

a) **for class j**

$$UA_i = \frac{x_{ii}}{\sum_{i=1}^{N_c} x_{ij}} = \frac{p_{ii}}{r_i}$$

b) **for combined classes**

$$UA_{tot} = \frac{1}{N_c}\sum_{i=1}^{N_c} UA_i = \frac{1}{N_c}\sum_{i=1}^{N_c} \frac{p_{jj}}{r_j}$$

4. Method of Hellden (H)

- Estimate of the "mean accuracy" -- the probability that a randomly chosen pixel of a specific class (k) will be properly classified
- This index is developed heuristically and cannot be derived on either a probability or mathematical basis.

$$H_k = \frac{2x_{kk}}{\sum_i x_{ik} + \sum_j x_{kj}} = \frac{2p_{kk}}{r_k + c_k}$$

5. Method of Short (S)

- Also called "mapping accuracy"
- S = 0 for no positive matches; S = 1 for perfect agreement.
- Not affected by sample size

$$H_k = \frac{x_{kk}}{\sum_i x_{ik} + \sum_j x_{kj} - x_{kk}} = \frac{p_{kk}}{r_k + c_k - p_{kk}}$$

6. Method of Cohen: coefficient of agreement, $\hat{K}$

- the proportion of agreement after chance agreement is removed from consideration.

$\hat{K} = 0$ when obtained agreement equals chance agreement.

$\hat{K} = 1$ for perfect agreement.

$K < 0$ if obtained agreement is less than chance agreement.

$$\hat{K} = \frac{p_0 - p_c}{1 - p_c}$$

p_o = proportion of correctly classified pixels

p_c = proportion of pixels correctly classified by chance

For individual classes:

$$\hat{K}_k \frac{N\sum_i x_{ii} - \sum_i \left[\sum_j x_{ij} \sum_j x_{ji} \right]}{N^2 - \sum_i \left[\sum_j x_{ij} \sum_j x_{ji} \right]} = \frac{p_{kk} - r_k c_k}{c_k - r_k c_k}$$

Reference (Test) Data

Classified data		water 1	urban 2	HDres 3	LDres 4	forest 5	grass 6	field1 7	field2 8	Total
unclass.	**0**	147	0	3	5	10	15	1	35	216
water	**1**	652	0	0	0	0	0	0	0	652
urban	**2**	1	231	61	11	1	2	0	0	307
HDres	**3**	0	44	465	247	0	27	0	0	783
LDres	**4**	0	0	8	587	205	15	16	0	831
forest	**5**	0	0	0	21	703	4	0	51	779
grass	**6**	0	0	4	3	1	345	0	1	354
field1	**7**	0	7	59	259	0	88	111	0	524
field2	**8**	0	0	0	3	8	45	2	357	415
	Total	800	282	600	1136	928	541	130	444	4861
									p0 =	3451

		Class accuracy estimates								all
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**classes**
Overall		-	-	-	-	-	-	-	-	-	0.71
Producer's		-	0.82	0.82	0.78	0.52	0.76	0.64	0.85	0.80	0.75
User's		-	1.00	0.75	0.59	0.71	0.90	0.97	0.21	0.86	0.75
Kappa		-	0.79	0.81	0.73	0.42	0.71	0.61	0.84	0.79	0.66