4.0 Normalization and transformation of features vectors

4.1 Normalization and visualization of features

In this chapter, measured values or rather the thereby derived data are to be represented by a m-dimensional feature-vector.

$$x = \begin{Bmatrix} x_0 \\ x_1 \\ x_2 \\ . \\ . \\ x_{M-1} \end{Bmatrix} \qquad x^T = \{x_0, x_1, x_2, \ldots x_{M-1}\}$$

We have to repeat the calculation with vectors. What is the result of a vector multiplied with a transposed vector? Is this a matrix or an amount? And so on.

$$x \cdot x^T, \quad x^T \cdot x, \text{ what is } A^T \cdot x$$

$$(x - m)^T K^{-1} (x - m)^T$$

$$A, K \quad \text{are matrices}$$

The components of the features-vector may either be integers or real numbers. Images with characters will in general consist of grey values only. But sometimes there is even just black and white. In this case 0 represents white and 1 represents black. Using a 12 bit A/D converter for measuring leads to values ranging from -2048 to 2046 (integers). The feature-vector often consists of different kinds of physical data. In a single vector, there may for example be colors with values red, green and blue as well as other measured values.
For different kinds of features it makes sense to use specific number sets (number ranges). An example: Welding steal leads to voltages ranging from 0 V to 1 kV and to currents ranging from 1,8 kA to 1,9 kA.

PROBLEM

By default, features with higher values have a stronger impact on the computation of the classifier data as compared to attribute with smaller values. It is important to consider, how strong a feature impact on the classification is. This problem is solved by normalization. It is the objective of normalization to have all attributes have a similar impact on the classification data. The normalization achieves this with the help of the mean value and the variance.
One has to distinguish between the learning-set and the test-set. The data to be normalized is in the learning-set. Perhaps you need for the classification procedure a further test set the normalization you need has to be measured after the learning set.

Example: sample-set from N pattern:

$$\mu_k = \frac{1}{N} \sum_{i=1}^{N} X_{ki} \qquad k = 1,2,\ldots K$$

$$\sigma_X^2 = \frac{1}{N-1} \sum_{i=1}^{N} (X_{ki} - \mu_k)^2$$

Normalization: $$X_{ki} = \frac{X_{ki} - \mu_k}{\sigma}$$

After the normalization the features have a mean value of zero and a variance one. There are also other possibilities to manipulate the components impact on the classification. Sometimes it is useful

- with respect to the number range that is actually being used or
- with respect to the actual feature spread

In this domain, there are lots of different techniques in use. For example you may just confine the largest attribute values or just cut them. One should also be able to build subsets of feature ranges, classify the learning set and (if possible) classify the test set. In case that you need the covariance matrix, then make sure that very small numbers are replaced by 0. This will increase the stability of the covariance matrix.

- **feature visualization**

It is very useful to represent features graphically. In many cases, a developer of a classification system can determine the features impact on classification just by looking at a graphical representation.

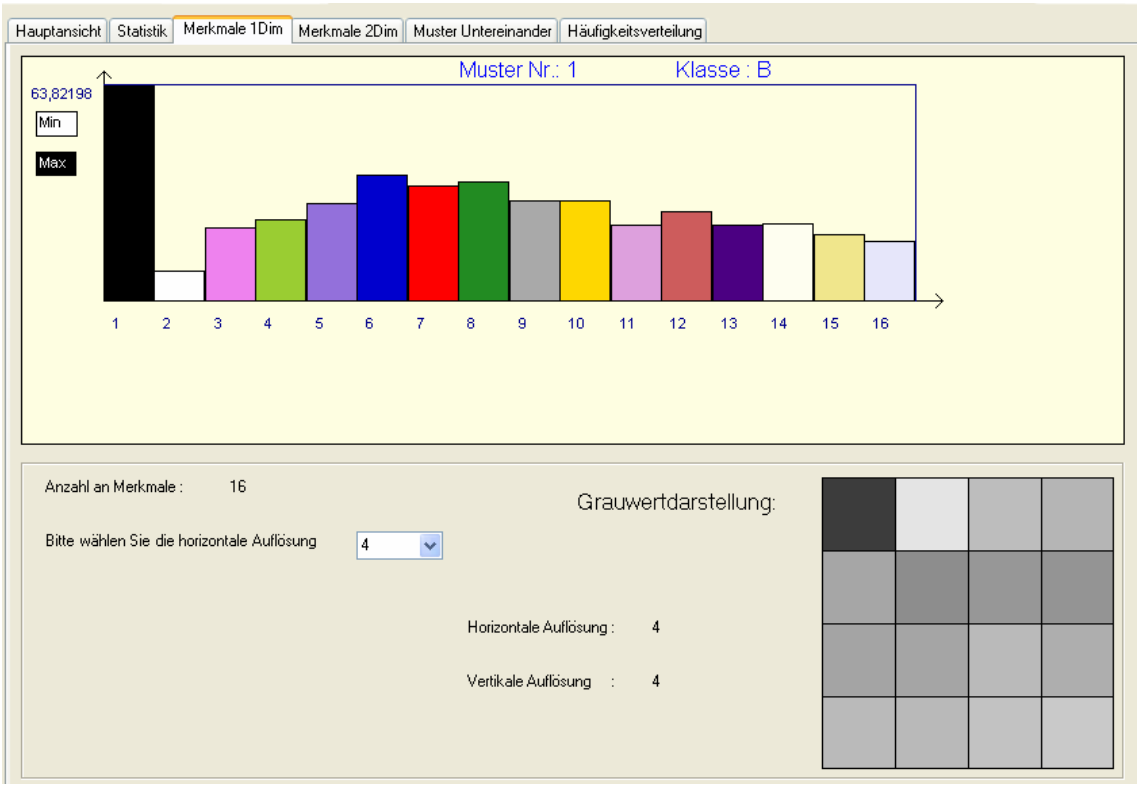
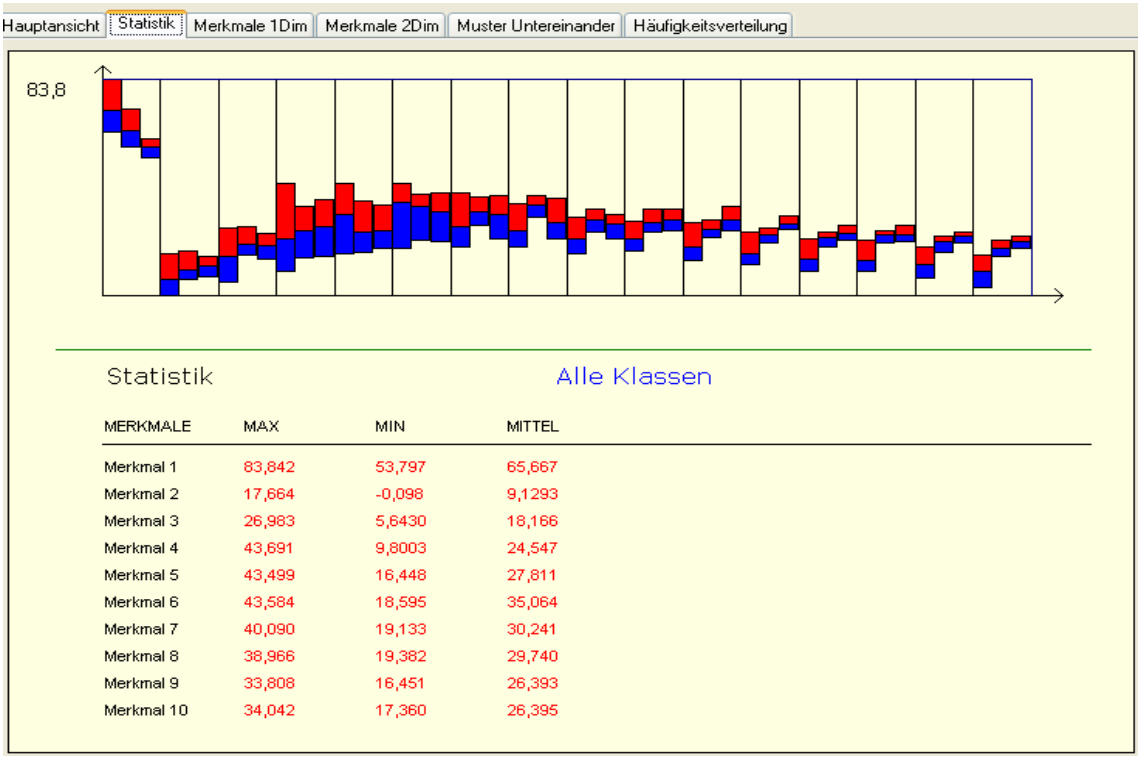


Figure 4.01: Feature vector with 16 components.



MERKMALE	MAX	MIN	MITTEL
Merkmal 1	83,842	53,797	65,667
Merkmal 2	17,664	-0,098	9,1293
Merkmal 3	26,983	5,6430	18,166
Merkmal 4	43,691	9,8003	24,547
Merkmal 5	43,499	16,448	27,811
Merkmal 6	43,584	18,595	35,064
Merkmal 7	40,090	19,133	30,241
Merkmal 8	38,966	19,382	29,740
Merkmal 9	33,808	16,451	26,393
Merkmal 10	34,042	17,360	26,395

Figure 4.02: Sixteen features with lower limit-value, upper limit-value, and mean value

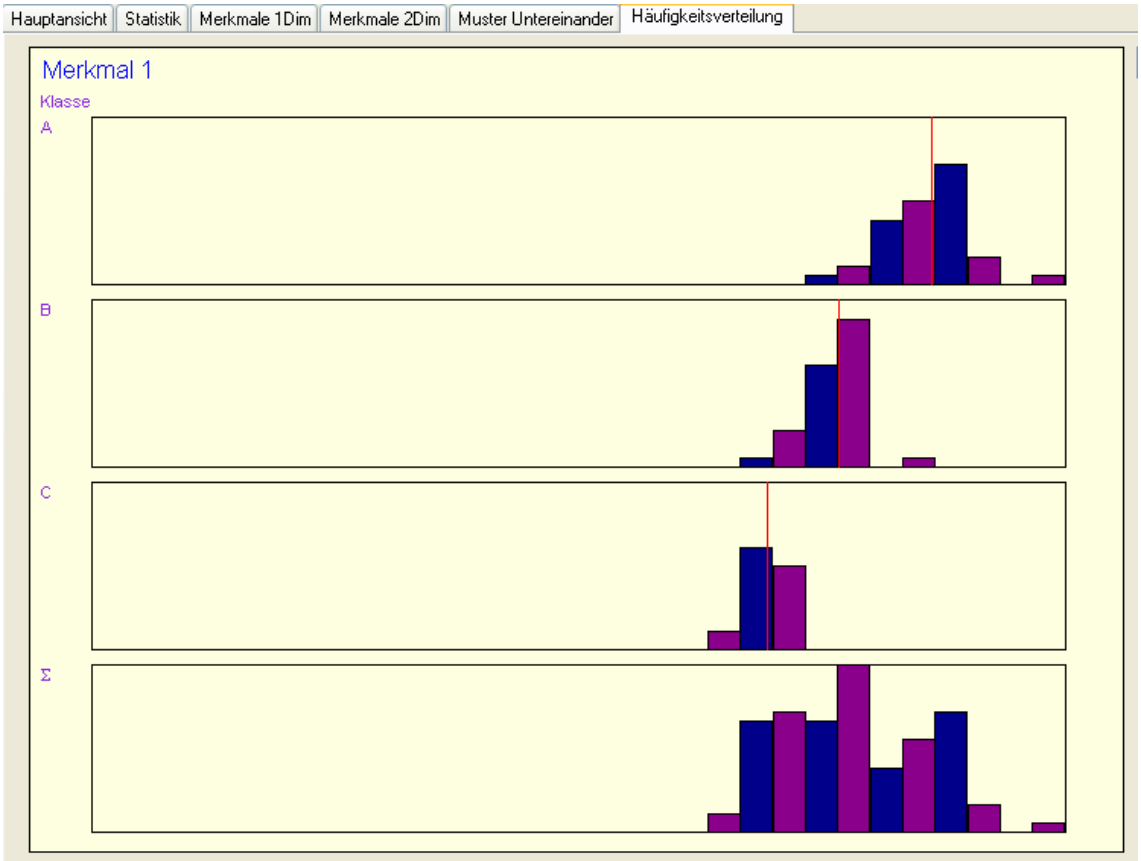


Figure 4.03: Feature 1. Classes A, B, and C. The red line stands for the mean value. The mean value and the spreads of the features within the classes are indications for relationships between features.

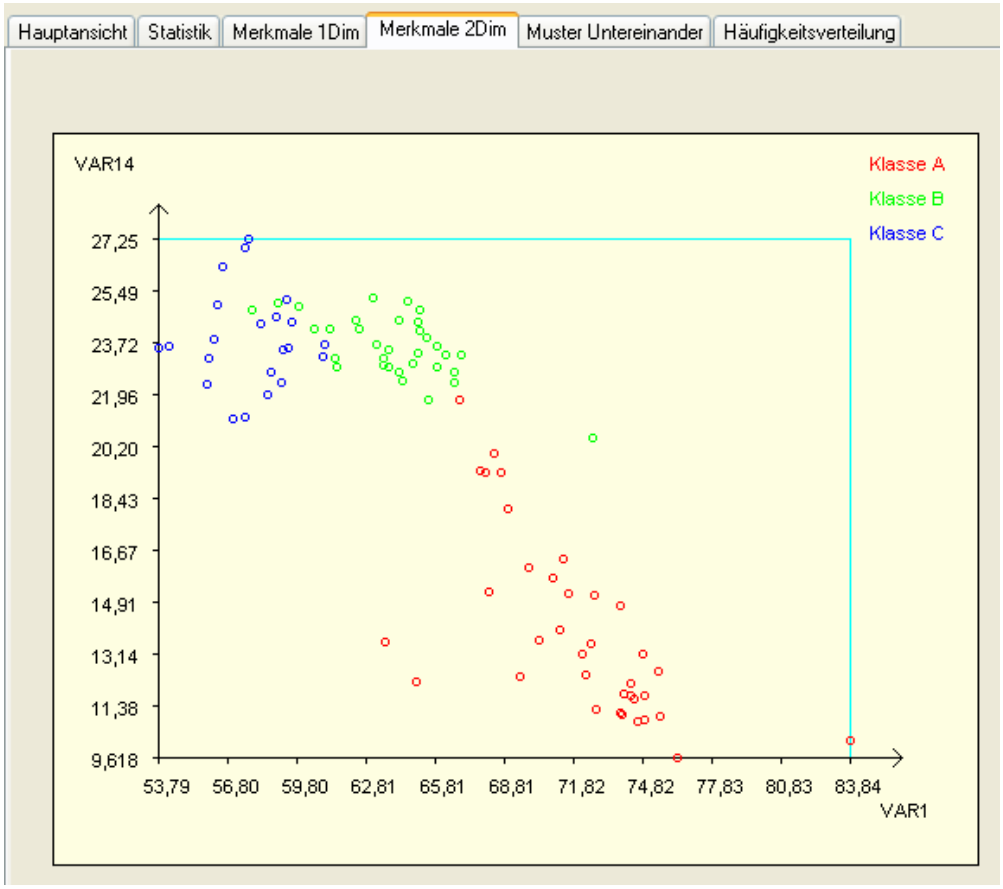


Figure 4.04: Two dimensional visual representation of feature 1 and feature 14. Classes A, B, and C

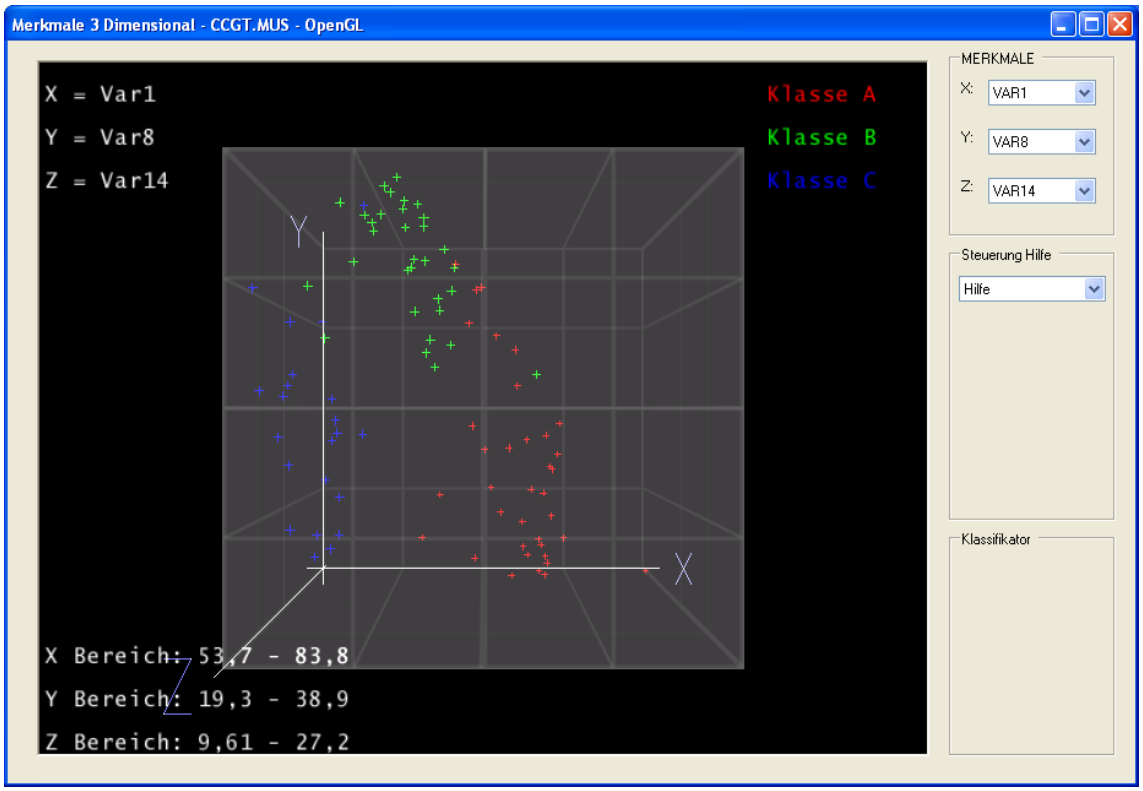


Figure 4.05: Three dimensional visual representation of feature 1, feature 8, and feature 14. Classes A, B, and C

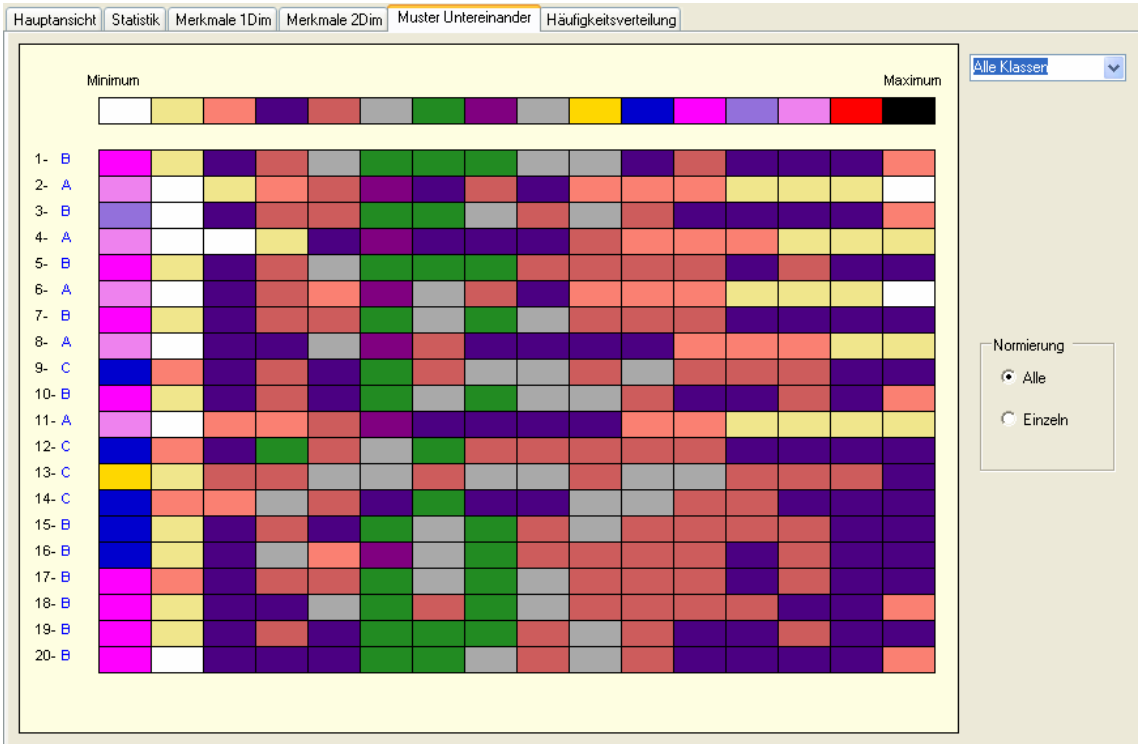


Figure 4.06: You see 20 pattern of all classes B, A, B and so on.
Each value is represented by a colour. The minimum value is white, the maximum value is black.

If you only allow class A, then you can see outliers.

4.2 Probability of occurrence, outlier test

Almost all classification procedures aim at producing classes, where the number of patterns per class is approximately equal for all classes. Should the number of patterns significantly vary, then there are two common techniques to overcome this: multiple reuse of patterns and the generation of new patterns based on existing patterns. New attributes, that are to be combined to a pattern vector, are determined as follows:

$$x' = r \cdot var(x) \cdot rand + x$$

r = factor of change
var(x) = spread of attribute x,
rand = randomly chosen number in a range from -1 to +1.

Sure it is better to have a bigger number of pattern vectors instead.

- **Outlier Test**

There is already a big amount of articles about this subject. The learning sample has to be verified before computing the classifier data.
Simple solution: Build a tolerance band around the mean value

r = factor of change
ub = upper boundary $ub = \mu + r\mu$
lb = lower boundary $lb = \mu - r\mu$

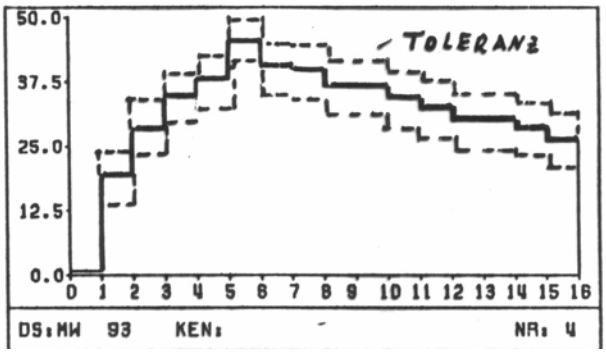


Figure 4.07: Mean value and tolerance band

A simple test determines how many patterns and how many attributes there are outside the tolerance band. Based on this information, the developer of the classification system has to decide, whether the pattern has to be removed and whether or not certain features are neglect able and may be skipped.

- **Comparison of a learning set and a test set**

A classifier for testing the correct function of electronic motors works rejection quota of 5% [Becker85]. After applying the classifier, specialized testing experts examined the rejected motors once more very thoroughly. Way later, at time t2, the rejection quota increased to 15%. So one decided to take a new test sample, examined it again by the specialized testing experts and compared the results with the learning sample.

Figure 4.08 shows the mean vectors of class A, B, and C, the total mean vector for the learning set and the test set. The y axis is used for the level and the x axis shows attribute numbers (frequency axis). The gap between learning sample and test sample is highlighted by a hatch.
The gap becomes very obvious in class B. But also with the total medium vector it becomes obvious that at time t2 (generation of the test sample) the attributes 2 and 4 showed values, that significantly vary from those measured at t1 (generation of the learning sample).

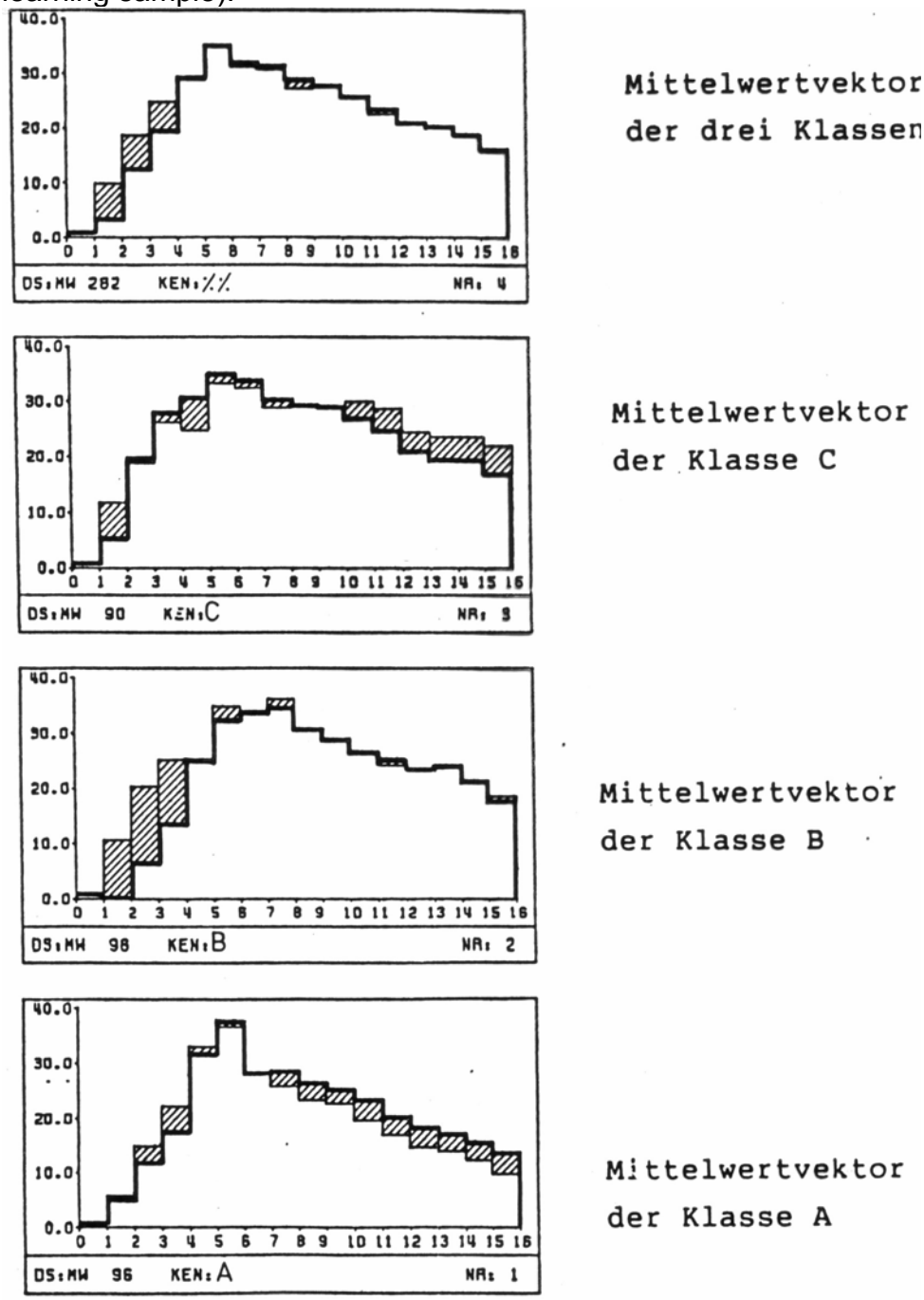


Figure 4.08: Mean vectors of class A,B,C and whole mean value of the learning set and test set.

This allows us to understand the strong increase of the rejection rate. No matter what the classification procedure looks like, the mean vectors of the learning set should only slightly differ from those of the test set.

4.3 Feature evaluation and feature extraction

- Distance measure (DiM)

Each feature x_γ is evaluated by the quality factor G without consideration of the other features with a distance measure G_γ. A requirement for this is the statistic independence of the features. This requirement is often not fulfilled, and nevertheless the quality factor is used.
For the calculation of G_γ the class mean values $\mu_{k\gamma}$ and dispersions $\sigma^2_{k\gamma}$ (for k = 1, 2) of the features x_γ are needed. Estimated values of both numbers can be achieved from the learning set. The definition of the quality is:

$$G_\gamma = \frac{(\mu_{1\gamma} - \mu_{2\gamma})^2}{(\sigma^2_{1\gamma} - \sigma^2_{2\gamma})}$$

With a bigger difference of the mean values on one hand and a smaller sum of the variances on the other hand, the value G_γ is getting bigger and the separation in to two classes with the characteristic c_γ is better possible. This can be seen in figure 4.09, in which the distribution densities $w(c_\gamma)$ of two features with good and bad separation effectiveness are represented [Niemann83].

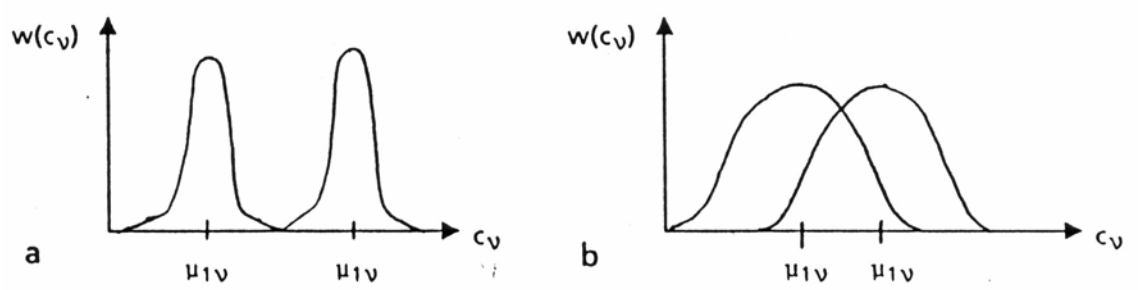


Figure 4.09: Separation-effective (a) and separation-ineffective (b) feature and its influence on the distance measure (DiM)

- **Branch and Bound algorithm (BBA)**

To select the best subset with m features from a quantity of n features would require the classification with all possible subsets or at least the calculation of a criterion for all subsets. With increasing n more calculation is necessary and very soon it exceeds an acceptable effort.

With the Branch and Bound algorithm (BBA) [Narenda77] the optimal subset can be achieved without long calculations. The only condition is that the criterion J, which describes the separation ability of the features, is a monotone function; it is assumed that a subset cannot ensure the separation of the features better than the quantity of features, which contains the subset.

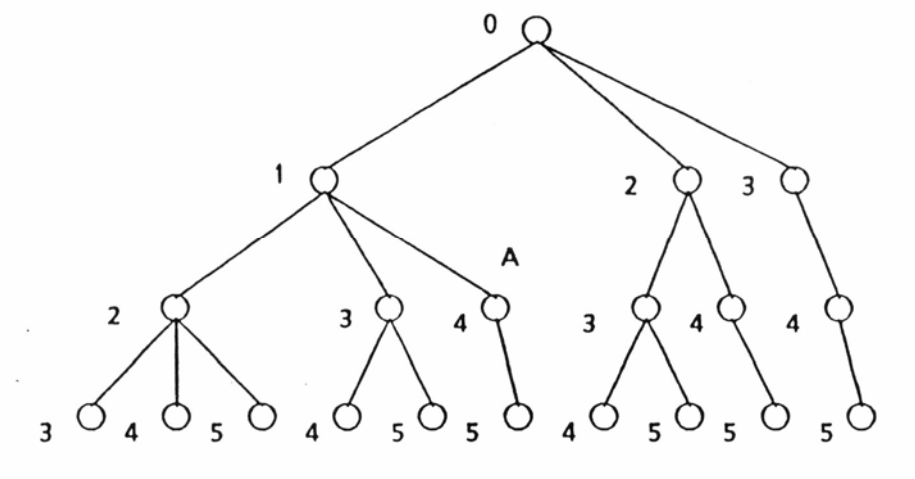


Figure 4.10: Branch-and-Bound algorithm

Initially for the algorithm the p = n - m features x_1..., x_k..., x_p, which are not further considered, are arranged arbitrarily by k = 1 to k = p with increasing k.
With a tree all possible sequences can be illustrated.
Figure 4.10 shows the tree for m = 2 and n = 5. Each knot in the tree is already characterized by the k numbers of the features, which were put aside up to the knot; for example knot A is characterized through (1,4). Further each knot can be assigned to $Jk(x_1,...,x_k)$, which is calculated from the remaining n - k features.
The goal is to find the features x_1^*,…, x_p^* for which the criterion J is a maximum value $J_p(c_1^*...,x_p^*)$.
The algorithm starts with the calculation of a subset $J_p(c_1..., c_p)$, according to a branch in the tree, and this gives the lower limit $B = J_p(c_1,...,c_p)$.
Now successively $J_1(c_1)$, $J_2(c)$, etc. are calculated until $J_k \leq B$, because then all following J_k are smaller due to monotony of J and it is not necessary to calculate them. Starting with the paths, which contain the most branching, the number of calculations can be reduced.
The algorithm in the program, which is used in this work, needs a learning set, which contains feature vectors of two or more classes. It offers the Euclidean distance as criteria J

$$d_k^2(x,\mu_k) = (x-\mu_k)^T(x-\mu_k)$$

and the Mahalanobis distance.

$$d_k^2(x) = (x-\mu_k)^T K_k^{-1}(x-\mu_k)$$

between the mean value vectors μ and x of the features vectors of two classes [Narenda77]. At the following investigations the Euclidean distance is selected, since the mean distance classifier uses this distance measure (K_k^{-1} is the inverse covariance matrix off class k).

- **sequence list at the least square method classifier**

The calculation of the classifier data of the least square method classifier is done in several steps. In each calculation step these component of the features vectors of the learning set are accepted in the order of their importance to the estimation equation. The feature, which gives the largest contribution for the completion of the optimization criterion of the least square method classifier, is first accepted. During the calculation of the classifier data the sequence list about the importance of the features is obtained. In chapter 5 figure 5.09 you will learn more about the sequence list and the admission of the features.
The statements always apply to the present learning set. A learning set with 282 patterns (3 classes A, B and C, 16 features) is given. In figure 4.11 the five most important features are indicated from the three presented procedures.

Feature	DiM	BBA	LMC
1			
2	2		
3			
4	3	4	5
5			
6			
7			
8			
9			
10			
11			
12			
13	4	3	4
14	1	2	1
15		1	2
16	5	5	3

Figure 4.11: The most important features after the distance measure (DiM), Branch and Bound algorithm (BBM) and the least square method classifier (LMC)

These results were achieved in a study thesis and should be repeated at a larger learning set, with more patterns.

4.4 Principal component analysis (PCA)

The principal component analysis is a procedure, which reduces the correlation between the features. As a consequence the number of features, important for the classification, can be reduced.
The correlation of the features is analyzed with the covariance matrix. The variances of the features itself are on the diagonals of the covariance matrix, the variances among of the individual features are on the none diagonals.
Among themselves one calls the variances the covariances.
The given coordinate system is by the principal component analysis shifted and turned using the covariance matrix and the eigenvalue and eigen vector system, and the features are no longer correlated among themselves.
Thus we win a sentence of new features, which are **uncorrelated** from the old features.
The most important features are the largest eigenvalue or the largest variance. Features with a smaller variance do not contribute anything to the reconstruction.
In the following example two-dimensional feature vectors are given. Class1 is marked by * and class 2 by +.

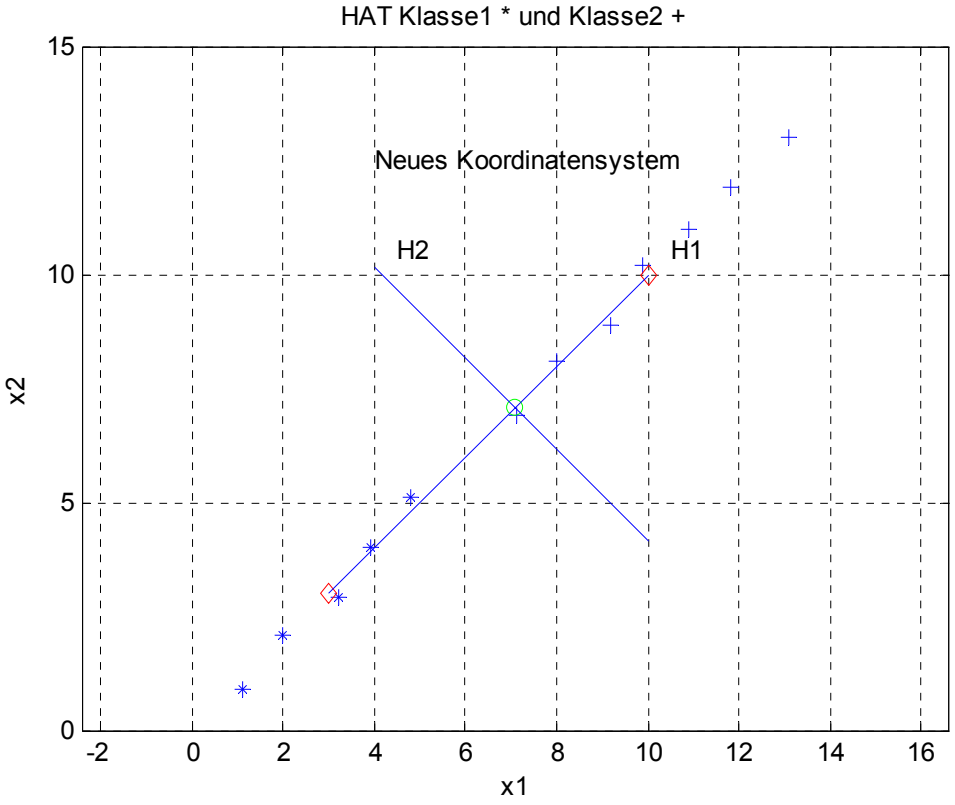


Figure 4.12: With the given coordinate system x1, x2 both features are needed for the separation. With a new coordinate system H1 and H2 into the centre only H1 is sufficient to separate the classes. (H1 stands for centreline one H2 for the axis two normal on H1)

The principal component analysis is calculated by the first two statistic moments - the expected value μ and the covariance matrix K – of the sample-producing process. There is no distinction after classes, always the total process is taken into account.

$$\mu = \{E(x)\},$$
$$K = E\{(x-\mu)(x-\mu)^T\}$$

For the calculation the eigenvalue problem has to be solved with the covariance matrix. This gives a pair of eigenvalues $\boldsymbol{\lambda}_m$ and eigen-vectors $\mathbf{b}_m$, which are arranged according the descending size of the eigenvalues. The most important mathematical connections are represented in the two equations

principle component analysis $w = B^T(x-\mu)$

back transformation $\tilde{x} = B \cdot w + \mu$

B rectangular N M matrix of the first eigen-vectors arranged according to the size. The eigenvalue vectors are orthogonal. (M number of the eigen-vectors, N number of the features of the vector x)
The reconstruction error R^2 can be calculated directly from the eigen-values λ_m

$$R^2 = E\{|x-\tilde{x}|^2\} = \sum_{m=M+1}^{N} \lambda_m$$

Figure 4.13: Original picture above with the dimension = 256
reconstructed with 20, 25, 30, 35 and 40 eigen-values [Schuermann96].

Other name for PCA is some times “Karhunen Loéve transformation”. The principal component analysis is frequently used.