# Learning Vector Quantization for Multiclass Classification: Application to Characterization of Plastics

Gavin R. Lloyd,[†] Richard G. Brereton,*,[†] Rita Faria,[‡] and John C. Duncan[‡]

Centre for Chemometrics, School of Chemistry, University of Bristol Cantocks Close, Bristol BS8 1TS, United Kingdom, and Triton Technology, Ltd., 3 The Courtyard, Main Street, Keyworth, NG12 5AW, United Kingdom

Received January 19, 2007

Learning vector quantization (LVQ) is described, with both the LVQ1 and LVQ3 algorithms detailed. This approach involves finding boundaries between classes based on codebook vectors that are created for each class using an iterative neural network. LVQ has an advantage over traditional boundary methods such as support vector machines in the ability to model many classes simultaneously. The performance of the algorithm is tested on a data set of the thermal properties of 293 commercial polymers, grouped into nine classes: each class in turn consists of several grades. The method is compared to the Mahalanobis distance method, which can also be applied to a multiclass problem. Validation of the classification ability is via iterative splits of the data into test and training sets. For the data in this paper, LVQ is shown to perform better than the Mahalanobis distance as the latter method performs best when data are distributed in an ellipsoidal manner, while LVQ makes no such assumption and is primarily used to find boundaries. Confusion matrices are obtained of the misclassification of polymer grades and can be interpreted in terms of the chemical similarity of samples.

## 1. INTRODUCTION

Classification has an important role in pattern recognition, especially for predictive modeling.[1] There are numerous approaches for determining classes between groups of samples in the chemical literature developed over 40 years. Many approaches such as partial least-squares−discriminant analysis (PLS-DA)[2−6] are linear methods, which involve modeling data sets using linear projections or boundaries. Methods such as the Mahalanobis distance[7−9] assume that the probability of a sample belonging to a class diminishes as it gets farther away from a weighted centroid. There are many variations on this theme, for example, using Gaussian density functions.[10,11] One problem with these approaches is that they do not take into account what are often complex, nonlinear boundaries between classes. In many forms of complex data, the boundaries, whereas possible to define effectively, cannot necessarily be approximated easily by straight lines, circles, or ellipsoids, and as such, many of the classical approaches to classification do not perform effectively.

A recently developed alternative that has had much success in biology, where complex nonlinear problems are common, is support vector machines (SVMs).[12−17] SVMs attempt to find complex nonlinear boundaries between classes, usually using the so-called "kernel trick", which transforms a nonlinear problem in a small number of dimensions into a linear problem in higher dimensions. Whereas SVMs are becoming increasingly widespread, they were originally developed for two class problems, where samples are assigned either to one group or another. They can be extended to multiclass situations,[17−20] but the implementation is very awkward, and when there is a very large number of groups, it can be quite complex. In addition, there are problems optimizing SVMs, which can be tackled when there is a small number of classes, but which become computationally very intense if there are several groups. Learning vector quantization (LVQ)[21−23] is a method that has recently been reported in the machine learning literature and is an alternative approach to multiclass boundary problems. The principle is based on a somewhat simpler and more historic approach of k-nearest neighbors (kNN)[8,9] in which a sample is assigned to a group by determining the distance (in multidimensional space) of a sample to members of a training set, choosing the "majority vote" for classification. Whereas kNN is computationally straightforward, there are several limitations. The first is that there is an underlying assumption of approximately equal numbers of samples in each group; otherwise, the voting may be biased. The second is that some groups may be more dispersed than others, so the significance of a large distance in a more dispersed group may be less than for a tighter group. A third and quite serious issue involves the validation of predictive models. When using predictive kNN models on a training set, including the sample to be predicted in the autopredictive model will bias the results, as an autopredictive model of $k = 1$ will always result in perfect classification. Ideally, this sample should be excluded, but then the autopredictive models for each sample will be based on a slightly different training set. For example, if there are 10 samples for a given class in a training set, then there will be 10 different autopredictive models, each consisting of nine samples; this then causes a dilemma when extending these models to a test set because each of the nine

* Corresponding author. E-mail: r.g.brereton@bris.ac.uk.
[†] University of Bristol Cantocks Close.
[‡] Triton Technology.

**Table 1.** Polymer Grades Used for this Analysis

| | polymer group | supplier | polymer grade | number of samples |
|---|---|---|---|---|
| amorphous | polystyrene (PS) | Polimeri | SR(L)550 | 9 |
| | polystyrene (PS) | Polimeri | R850E | 9 |
| | polystyrene (PS) | Polimeri | N2560 | 7 |
| | polystyrene (PS) | Nova Chemicals | 338 | 10 |
| | acrylonitril−butadiene−styrene (ABS) | Polimeri | F332 | 13 |
| | acrylonitril−butadiene−styrene (ABS) | Polimeri | B432/E | 15 |
| | acrylonitril−butadiene−styrene (ABS) | unknown | JG | 9 |
| | acrylonitril−butadiene−styrene (ABS) | BASF | Terluran GP-35 Natural | 10 |
| | polycarbonate (PCarb) | Dow | Calibre 200 | 10 |
| semicrystalline | low-density polyethylene (LDPE) | Borealis | FA3223 | 7 |
| | low-density polyethylene (LDPE) | Borealis | FB4230 | 9 |
| | low-density polyethylene (LDPE) | unknown | 742gr | 10 |
| | low-density polyethylene (LDPE) | unknown | Metler | 10 |
| | low-density polyethylene (LDPE) | Shrivatsa | Shrivatsa s1 | 10 |
| | low-density polyethylene (LDPE) | Shrivatsa | Shrivatsa s2 | 10 |
| | polypropylene (PP) | Borealis | HA507MO | 9 |
| | polypropylene (PP) | Borealis | RD204CF | 7 |
| | polypropylene (PP) | Borealis | HL512FB | 9 |
| | polypropylene (PP) | Borealis | BA212E | 10 |
| | polypropylene (PP) | Borealis | KSR4525 | 10 |
| | high-density polyethylene (HDPE) | Borealis | FS1470 | 10 |
| | high-density polyethylene (HDPE) | Sabic | B5823 | 10 |
| | high-density polyethylene (HDPE) | Aqualita | Padmex 60200 | 10 |
| | polyamide6 (PA6) | DSM | Akulon K123 batch 2 | 10 |
| | polyamide6 (PA6) | DSM | Akulon K123 batch 1 silver | 10 |
| | polybutylene terephthalate (PBT) | DSM | Arnite T04 200 | 10 |
| | polyethylene terephthalate (PET) | Wellman | Wellman | 10 |
| | polyethylene terephthalate (PET) | Eastman | Eastman | 10 |
| | polyethylene terephthalate (PET) | Dupont | 5143 | 10 |
| | polyethylene terephthalate (PET) | Dupont | 5154 | 10 |
| | | | total | 293 |

sample models results in different test set predictions. In this way, kNN differs from approaches such as PLS-DA or the Mahalanobis distance where a single statistical model is developed on the training set which can then be tested for autopredictive accuracy. Hence, it is not possible to directly compare the percent correctly classified in kNN to that in PLS-DA or many other established approaches, and there can be problems in using some approaches such as the bootstrap[24−26] for optimizing the model. LVQ overcomes these problems. Instead of using raw samples, a set of codebook vectors is obtained in which the vectors are characteristic of each class. The number of codebook vectors can be set as equal for each class, thus overcoming the problems of class size, and since these do not correspond to specific samples, all methods for optimization and validation can be employed in a directly analogous manner to most other approaches for modeling. LVQ does not assume a specific distribution of samples and, like most boundary methods, can be employed when there are very differently shaped classes. Finally, the codebook vectors are used to define boundaries between classes (as described below) and so allow a genuine multiclass boundary method.

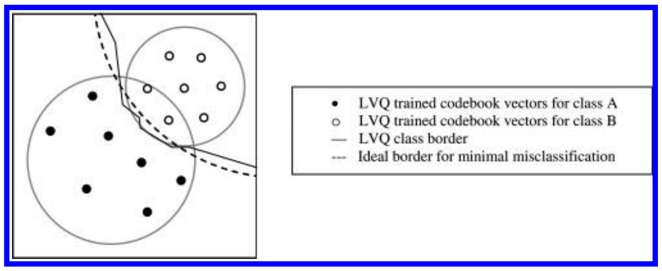
This paper illustrates the application of LVQ for the characterization of commercial polymers[27−29] from nine different groups. In turn, each group is usually characterized by several grades. Although it is not the purpose of this paper to distinguish grades, it is possible to interpret misclassifications according to grade as some grades contain components in different proportions while retaining the basic properties characteristic of their polymer group. LVQ is compared to a more established method which can also be employed when there are several groups, the Mahalanobis distance.

## 2. EXPERIMENTAL SECTION

**2.1. Dynamic Mechanical Analysis (DMA).** DMA is a common technique used to provide information about the mechanical properties of any viscoelastic material.[30] By applying an oscillating force and measuring the resulting displacement, the stiffness of the sample can be determined as well as other parameters such as the damping factor (tan $\delta$) and various moduli. For this analysis, the loss modulus ($E''$) was used, which represents the energy dissipated within the sample per oscillation. Full experimental details have been described previously.[28]

The temperature range in these studies was from −51 °C until the minimum stiffness of the polymer was reached, after which no further meaningful data could be collected and the DMA machine stops automatically. Measurements were made approximately every 1.5 °C. In total, each curve consisted of between 99 and 215 data points dependent on the highest recordable data point of the polymer. The data set consisted of a total of 293 analyses, originating from nine main amorphous or semicrystalline polymer groups.[31,32] Each group contained a number of different grades with the exception of polycarbonate (PCarb) and polybutylene terephthalate (PBT), both of which only contained one grade. Table 1 contains a full listing of polymer groups and grades used in this analysis.

**2.2. Software.** The control of DMA and data acquisition was performed using software written by Triton Technology, Ltd. in Visual Basic which operates under Excel. Subsequent



● LVQ trained codebook vectors for class A
○ LVQ trained codebook vectors for class B
— LVQ class border
--- Ideal border for minimal misclassification

**Figure 1.** Comparison of LVQ-defined class border and the ideal border for minimum misclassification rate.

data processing and visualization were performed using in-house routines in Matlab version 7 (R14; The MathWorks Inc., Natick, Massachusetts).

## 3. DATA ANALYSIS

**3.1. Preprocessing.** Because the measurements for different polymer samples were not performed at the same equally spaced temperatures $\theta$, the data were linearly interpolated according to eq 1:

$$x = \frac{(\theta - \theta_1)(x_2 - x_1)}{(\theta_2 - \theta_1)} + x_1 \qquad (1)$$

where $x$ is the new, interpolated $E''$ value between temperatures $\theta_1$ and $\theta_2$, and $x_1$ and $x_2$ are the recorded measurements at these temperatures. The starting temperature for each of the polymers was $-51$ °C, and missing values at the end of the data were replaced with the last recorded value. Each sample vector contained 215 data points corresponding to an interpolated or added $E''$ value for each of the temperatures between $-51$ and $+270$ °C in increments of 1.5 °C. The samples were then combined into a matrix $\mathbf{X}$ of dimensions $293 \times 215$ ($= I \times J$), where each row corresponds to a sample and each column to values of $E''$ at an interpolated temperature $\theta$. The data matrix $\mathbf{X}$ was then used to train and validate the LVQ procedure for classifying the polymer samples by dividing it into two sets: a training set and a test set. The training set was used to train the LVQ algorithm, and the test set, whose samples were not used to build the model, was used to test the performance of the method as described in section 3.4.

**3.2. Learning Vector Quantization (LVQ).** Learning vector quantization is a two-layer neural network that uses supervised learning for multiclass classification.[21,23] Similar to self-organizing maps[33,34] and also closely related to kNN methods,[8,9] the purpose of LVQ is to define regions of the input space to which samples can be assigned with a minimized chance of being misclassified. Figure 1 shows the hypothetical distributions of two classes, A and B. The optimal class boundary is found at the minimum overlap between the two classes, where the proportions of each class being misclassified are smallest. The main issue is that, for real data, the distributions of the classes are not known, and so the optimal boundary cannot be determined directly. In LVQ, the optimal boundary is approximated by generating a set of points for each class called codebook vectors, as introduced above. The borders are modeled using the midplane between codebook vectors on either side of it. The LVQ algorithm then iteratively adjusts the position of the codebook vectors so that the borders defined are placed such

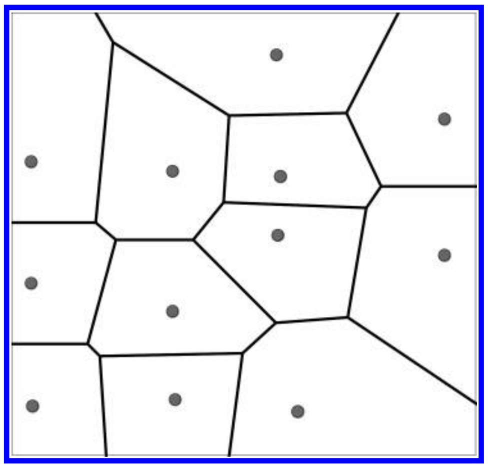


**Figure 2.** Voronoi tessellation of 12 points.

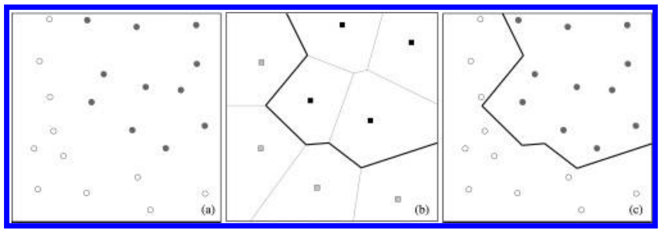


**Figure 3.** (a) Sample data (two classes, 22 samples). (b) Finalized codebook vector positions (points) and their Voronoi tessellation (lines). Class border is defined by bold lines. (c) Class border (lines) with respect to the classification of the sample data.

that the number of samples misclassified is minimized. In this way, the optimal borders can be approximated without knowing the distributions of the classes.

A useful concept for illustrating this is Voronoi tesselation.[35] Figure 2 represents a two-dimensional space in which each point represents a codebook vector. The space is divided up into regions so that each region contains one codebook vector. Any point falling in this region will have the codebook vector as its nearest neighbor. It is these lines and regions that make up the Voronoi tessellation. By removing lines that are between codebook vectors of the same class, the Voronoi regions can be merged so that the class region is represented. Any point falling into the defined class region will have a codebook from that class as its nearest neighbor. The lines remaining between codebook vectors from different classes therefore represent the decision border between classes.

Figure 3a represents 22 samples corresponding to two classes using two measurements. If a linear border is used to separate the two classes, then some misclassifications will occur. If LVQ is used, however, a piecewise-linear border can be defined by adjusting the positions of a number of codebook vectors so that misclassification rate is approximately minimized. Figure 3b shows the finalized positions of eight generated codebook vectors (four for each class) and the Voronoi tessellation for these points. Since the codebook vectors have been assigned to class regions, some of the lines (bold) can be used to distinguish between codebook vectors from different classes. The border defined by the codebook vectors can also be used to classify the samples (Figure 3c) as they were used to optimize the final positions of the codebook vectors so that there is a minimum

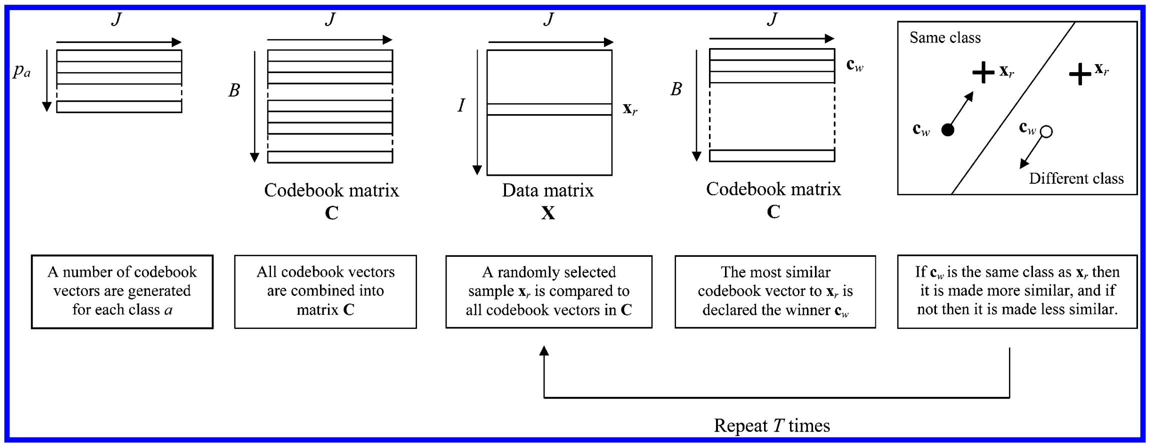

J
$p_a$
B
Codebook matrix **C**
I
$\mathbf{x}_r$
Data matrix **X**
$\mathbf{c}_w$
Codebook matrix **C**
Same class
Different class
A number of codebook vectors are generated for each class *a*
All codebook vectors are combined into matrix **C**
A randomly selected sample $\mathbf{x}_r$ is compared to all codebook vectors in **C**
The most similar codebook vector to $\mathbf{x}_r$ is declared the winner $\mathbf{c}_w$
If $\mathbf{c}_w$ is the same class as $\mathbf{x}_r$ then it is made more similar, and if not then it is made less similar.
Repeat *T* times

**Figure 4.** Schematic of the LVQ1 algorithm.

chance of misclassifying a sample. The border between classes therefore approximates the decision border without requiring the distributions of each class.

The number of initial codebook vectors for each class can have a significant effect on the classification ability of the LVQ algorithm. Using multiple codebook vectors per class can improve the description of the classification borders, but too many may cause overfitting and result in poor classification of the samples not included in the training set. The number of codebook vectors can be determined empirically (as was the case for this work) or by modifying the LVQ algorithm to add and/or remove codebook vectors where needed.[36,37]

The original (LVQ1) algorithm was first described by Kohonen,[21,22] who also suggested further improvements in the forms of LVQ2,[38] LVQ2.1,[38] and LVQ3.[21,22]

Often, the LVQ1 algorithm is sufficient to obtain satisfactory classification. Accuracy can generally be increased however by following the LVQ1 algorithm with one of the improved algorithms using a low initial learning rate in an attempt to fine-tune the location of the decision borders. In this paper, LVQ1 was followed sequentially by LVQ3, and so further discussion will be limited to these cases. The sequential combination of the two algorithms will be referred to as LVQ1+3.

The full method for validation is described in section 3.4.

*3.2.1. LVQ1.* The LVQ1 algorithm (Figure 4) is the simplest algorithm because the position of only one of the codebook vectors is updated at a time. Most improvements to the LVQ algorithm attempt to optimize the number of codebook vectors per class or make changes to the way in which the codebook vectors are updated. The basic algorithm, however, remains almost unchanged.

The first step is to generate the codebook vectors for each class in the training set that are then combined into a $B \times J$ matrix **C**, where $B$ is the total number of codebook vectors. The codebook vectors should initially be placed within their class region. This is to ensure that the codebook stays within its class region. If the codebook were to stray into the wrong class region, it may be that it is pushed away from the class in every iteration, causing it to never converge on an optimal position. Since the class regions are not yet known, one way to ensure the codebook vectors start within the correct class region is to use a sample that can already be correctly classified by kNN for example. For each iteration, $t$, which takes integer values from 1 to the maximum number of iterations, $T$, a learning rate $\alpha_t$ is defined by eq 2:

$$\alpha_t = \alpha_0\left(\frac{T-t}{T}\right) \tag{2}$$

where $\alpha_0$ is the initial learning rate.

The learning rate determines how much the codebook vectors are moved and hence how much the decision boundaries are adjusted in each iteration. The exact form of the function is not critical, provided that it reaches 0 over a suitable number of iterations. Recommended values[22] for $\alpha_0$ are to be no larger than 0.1, and for $T$, they are to be 50−200 times the number of codebook vectors. For this work, $\alpha_0$ was set to 0.09 and $T$ was set to 100 times the number of codebook vectors. Rather than attempt to select the optimum number of codebook vectors for each class, for this work, the numbers of codebook vectors per class were kept equal. This allowed classes of different sizes to be modeled and classified without bias. Since increasing the number of codebook vectors allows more complex borders to be defined, the entire validation procedure was carried out for an increasing number of codebook vectors per class from one to six to see if classification could be improved.

Once the number of codebook vectors has been determined, the positions of the codebook vectors are then iteratively updated for each value of $t$ according to the following steps.

1. A sample vector $\mathbf{x}_r$ is selected from the preprocessed $I \times J$ data matrix **X** (see section 3.1 for details), where $r$ is a randomly selected integer in the interval 1 to $I$ which is generated for each iteration.

2. The Euclidean distance between each of the codebook vectors and the selected sample is calculated:

$$d_{rb} = \sqrt{\sum_{j=1}^{J}(x_{rj} - c_{bj})^2} \tag{3}$$

3. The codebook with the smallest distance to the sample is declared the winner, $\mathbf{c}_w$, where $w$ is the index of the winning codebook in matrix **C** for each iteration. Depending on the class of the winner and the sample, the winner will be adjusted to try and improve the classification rate.

4. If the winning codebook is from the same class as the chosen sample, it is moved an amount proportional to the learning rate toward the selected sample:

$$\mathbf{c}_w = \mathbf{c}_w + \alpha_t (\mathbf{x}_r - \mathbf{c}_w) \quad (4)$$

If the winning codebook is associated with a different class from that of the sample, then it is moved a proportional amount away from the sample:

$$\mathbf{c}_w = \mathbf{c}_w - \alpha_t (\mathbf{x}_r - \mathbf{c}_w) \quad (5)$$

All other codebook vectors remain unchanged:

$$\mathbf{c}_b = \mathbf{c}_b \text{ for } b \neq w \quad (6)$$

When training of the codebook vectors is complete ($t = T$), the class borders are approximately located so that samples can be can be classified by assigning them to the same class as their nearest neighboring codebook vector with the minimum misclassification rate.

*3.2.2. LVQ3.* The basic LVQ1 algorithm can be modified to better approximate the class borders by updating two codebook vectors at a time, rather than just one.[22] Additionally, if the codebook vectors are from different classes, then they are only updated if the sample $\mathbf{x}_r$ falls into a window defined around the midplane of the winning codebook vectors.

Figure 5 shows how the window is defined about the midplane of the two winning codebook vectors in a two-dimensional case. The window is included to ensure that the class borders are adjusted such that the misclassification rate of samples on the edge of a class is at a minimum. If $d_{rk}$ is the Euclidean distance of the sample $\mathbf{x}_r$ from its $k$th nearest codebook, then the sample is considered to be inside a window of width $s$ relative to the sample dimensions if

$$\frac{d_{r1}}{d_{r2}} > \frac{1-s}{1+s} \quad (7)$$

This means that only codebook vectors with a sample near the midplane of the two codebook vectors will be updated, thus ensuring that the class border is defined most accurately where samples are close to the border. If the value of $s$ is too small, then the number of times a codebook vector is updated will be small, even for a large number of iterations. This means the codebook vectors have not been sufficiently trained, which results in poorer accuracy. A relative window width of 0.2−0.3 is generally recommended,[22] depending on the number of samples available for the learning process. For this work, a window width of 0.2 was used.

In LVQ3, there are two "winning" vectors, $\mathbf{c}_v$ and $\mathbf{c}_w$. Step 4 of the LVQ1 algorithm therefore needs to be updated for the codebook vectors to be adjusted appropriately as there are now four combinations for the class of the codebook vectors and the class of the sample that need to be considered (Table 2). If only one of the neighboring codebook vectors is from the same class as $\mathbf{x}_r$, then the rules applied are similar to those used in LVQ1. If $\mathbf{c}_v$ is the winning vector of the same class as $\mathbf{x}_r$, then it is moved toward the sample:

$$\mathbf{c}_v = \mathbf{c}_v + \alpha_t (\mathbf{x}_r - \mathbf{c}_v) \quad (8)$$

while $\mathbf{c}_w$, which is from a different class than $\mathbf{x}_r$, is moved away:

$$\mathbf{c}_w = \mathbf{c}_w - \alpha_t(\mathbf{x}_r - \mathbf{c}_w) \quad (9)$$

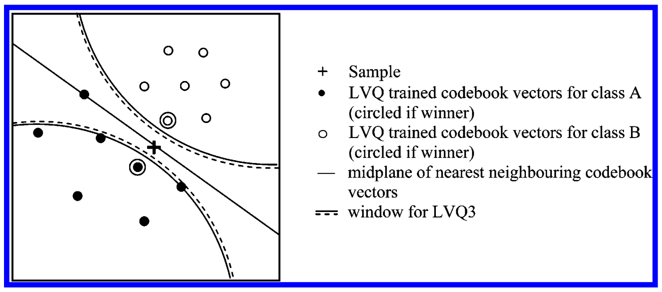

+ Sample
● LVQ trained codebook vectors for class A (circled if winner)
○ LVQ trained codebook vectors for class B (circled if winner)
— midplane of nearest neighbouring codebook vectors
window for LVQ3

**Figure 5.** Example of window used for LVQ3. Only if the sample is near to the midpoint of its two nearest neighboring codebook vectors will the codebook vectors be updated.

**Table 2.** Rules for Updating Codebook Vectors in LVQ3

| $\mathbf{x}_r$ class | $\mathbf{c}_v$ class | $\mathbf{c}_w$ class | update rule |
|---|---|---|---|
| A | A | A | move positions of $\mathbf{c}_v$ and $\mathbf{c}_w$ toward $\mathbf{x}_r$ |
| A | A | B | if $\mathbf{x}_r$ is within the window, then move the position of $\mathbf{c}_v$ towards $\mathbf{x}_r$ and the position of $\mathbf{c}_w$ away from $\mathbf{x}_r$ |
| A | B | A | if $\mathbf{x}_r$ is within the window, then move the position of $\mathbf{c}_v$ away from $\mathbf{x}_r$ and the position of $\mathbf{c}_w$ towards $\mathbf{x}_r$ |
| A | B | B | do not move any codebook vectors |

**Table 3.** Tuneable Parameters for LVQ1 and LVQ3

| parameter | recommended values | chosen value |
|---|---|---|
| $\alpha_0$ (LVQ1) | <0.1 | 0.09 |
| $\alpha_0$ (LVQ3) | <0.01 | 0.01 |
| $s$ | 0.2−0.3 | 0.2 |
| $\epsilon$ | 0.1−0.5 | 0.2 |
| $T$ | 50−200 × number of codebook vectors | 50 × number of codebook vectors |

The above rules may cause the codebook vectors to converge on a nonoptimal solution. To overcome this, an additional learning rule is implemented for when $\mathbf{c}_v$, $\mathbf{c}_w$, and $\mathbf{x}_r$ are all from the same class. The additional rule includes an extra learning factor $\epsilon$ to ensure that the codebook vectors converge on their optimal positions:

$$\mathbf{c}_u = \mathbf{c}_u + \epsilon\alpha_t(\mathbf{x}_r - \mathbf{c}_u) \quad (10)$$

where $\mathbf{c}_u \in \{v,w\}$. The exact value of $\epsilon$ is dependent on the window size, but generally it is advisable for it to have a value between 0.1 and 0.5.[22] LVQ3 is generally used after the LVQ1 algorithm to fine-tune the location of the codebook vectors. For this reason, the initial learning rate $\alpha_0$ for LVQ3 should start no higher than 0.01. In this work, a value of 0.2 was used for $\epsilon$ and $\alpha_0$ was initialized at 0.01. A table of tunable parameters and their values is shown in Table 3. In this paper, LVQ1 was always followed sequentially by LVQ3.

**3.3. Principal Component Analysis and Mahalanobis Distance.** In order to provide a comparison for the performance of the LVQ1+3 algorithms, the Mahalanobis distance measure was employed as an alternative classifier.[7−9]

The Mahalanobis distance measure is similar to the Euclidean distance measure except that it takes into account the correlation between variables and the dispersion (or variance) of each class by making use of the variance covariance matrix. The Mahalanobis distance ($d$) between the $i$th sample and the centroid of class A is defined by

$$d_{i\mathrm{A}} = \sqrt{(\mathbf{x}_i - \bar{\mathbf{x}}_\mathrm{A})\mathbf{C}_\mathrm{A}^{-1}(\mathbf{x}_i - \bar{\mathbf{x}}_\mathrm{A})'} \quad (11)$$

where $\mathbf{x}_i$ is the measurement obtained for the $i$th sample, $\bar{\mathbf{x}}_A$ is the centroid of class A, and $\mathbf{C}_A^{-1}$ is the inverse of the variance−covariance matrix of class A. In this method, the Mahalanobis distance between a sample and each of the class centroids is calculated, and each sample is then assigned to the class with the smallest distance.

There can be difficulties, however, when the number of samples is less than the number of measurements as the variance−covariance matrix does not have an inverse. Since the data set for this paper has 215 dimensions and some classes only contain 10 samples, a form of variable selection is required. Principal component analysis (PCA) is a well-known chemometric technique that can be used for variable reduction.[9]

An objective of PCA is to transform the data such that the maximum variance is projected onto each successive axis or PC (principal component). Successive components describe the variance orthogonal to the previous ones, which results in a number of components each of reduced variability. Since the later components describe only a small amount of the total variability, they can be safely discarded without losing useful information. In this way, PCA can be used to reduce the number of variables and, in this study, enables the Mahalanobis distance measure to be calculated. For other data sets, however, there may still be problems calculating the inverse of the covariance matrix if there are a number of highly correlated variables and the number of PCs to be retained is close to the number of samples.

The number of significant components is dependent on the features in a data set and not known in advance. In chemometrics practice, cross-validation,[39,40] especially the leave-one-out (LOO)[39] method, is commonly employed for reasons including the fact that it is commonly implemented in software and also that it is a rapid calculation. However, we have shown previously[3] that the bootstrap is more stable than LOO cross-validation. A likely explanation is that the bootstrap contains many iterations, so the optimal model is the average of many models. Especially when an optimum is fairly flat, but noisy, it is often difficult to be certain of the optimum using LOO cross-validation. Hence, in this work, we prefer the bootstrap.

Once the optimum number of PCs for classification has been determined, these were used to build a model on a training set and classify samples in a test set in a similar fashion to that used for LVQ1+3 except that samples are assigned to the nearest centroid as calculated by the Mahalanobis distance, rather than to the class of the nearest neighboring codebook vector. The full bootstrapping and validation procedure is described in section 3.4.

**3.4. Validation and Bootstrapping.** Validation is used to estimate the performance of a classifier by dividing the data into a training set and a test set.[25,26] Generally speaking, a model is built using the training set data. The test set samples can then be considered as "unknowns" as they were not used to build the model. Using the model to classify the test set "unknowns" therefore allows the performance of the model to be evaluated. Our strategy for validation has been discussed in more detail in the context of PLS-DA,[3] and in this paper, we summarize the main principles, as applied to the two approaches outlined above.

For LVQ1+3, the training set was used to optimize the location of the codebook vectors. These trained codebook vectors are then used to classify the test set, which can be considered as a collection of "unknowns" as they were not included in the training of the codebook vectors. The process is repeated a large number of times (>100), randomly selecting samples for the training set. An average "percent correctly classified" (%CC) can then be calculated to give an idea of performance, and step-by-step monitoring of the procedure means that the classification rate of individual samples can performed. For this work, approximately two-thirds of the numbers of samples in each class were randomly selected without replacement to create a 192 × 215 training set. The remaining 101 samples were used to create a test set to assess the classification ability of the LVQ1+3 algorithms whose learning was based on the training set. The entire process was repeated 200 times and an average classification rate calculated for both the training and test sets. A schematic of the steps taken during this analysis is shown in Figure 6.

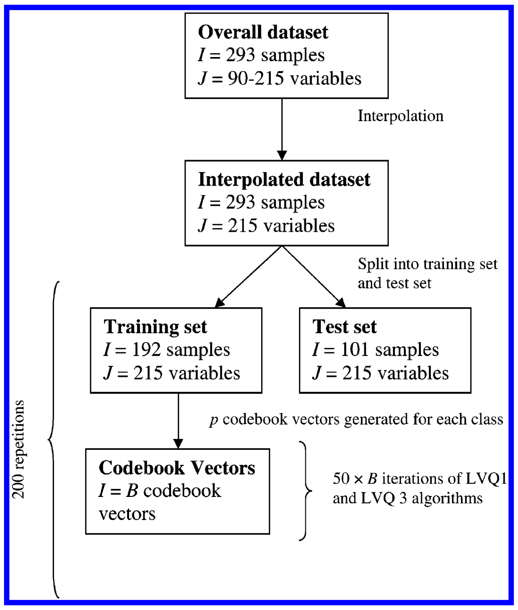


**Figure 6.** Schematic of data analysis steps for LVQ1+3.

For classification by Mahalanobis distance, the data is divided into a training set and a test in exactly the same way as for LVQ1+3. Before data can be modeled, the optimum number of PCs must first be selected. In order to do this, the training set was further split into a bootstrap set and a validation set. The bootstrap set consisted of samples selected at random with replacement from the training set. Class sizes were kept in the same proportions as in the training set to give a 192 × 215 bootstrap set. Any samples not chosen for the bootstrap set were used to create a validation set of approximately 70 × 215 (the number of samples varying for each iteration of the bootstrap). A model was then built on the bootstrap set and an increasing number of PCs used to predict the validation set samples. The modal number of PCs which gave the minimum classification error by Mahalanobis distance after 200 repetitions was then used to classify the test set data using a model built on the training set. The entire process was repeated 100 times and an average classification rate calculated for the test set. A schematic of the steps taken for bootstrapping and validation for the Mahalanobis distance method is shown in Figure 7.

Once validation is complete, the number of times a sample was selected for the test set and the number of times it was

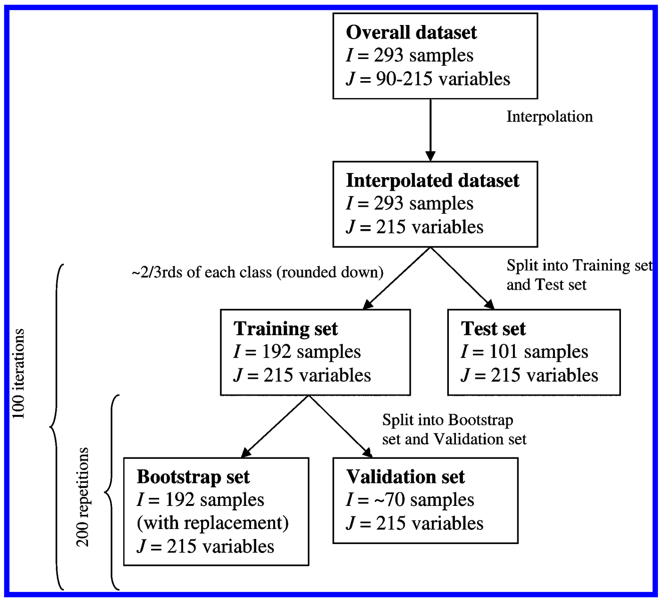


**Figure 7.** Bootstrapping and validation for classification by Mahalanobis distance.

correctly classified while in the test set can be used to create confusion matrices. For this work, there are two types of confusion matrices. The first is the percentage number of times a sample from a polymer group is classified as each of the groups, for example, the percentage number of times PS is classified as PS, ABS, PP, and so forth. This allows the overlap between similar polymer groups to be examined. The second type of confusion matrix used for this work shows the percentage number of times a sample from a specific grade is classified into a polymer group, for example, the percentage number of times Shrivatsa s1 is classified as PS, ABS, PP, and so forth. This allows the overlap between groups to be further explored by attributing the misclassifications to specific grades of polymer.

## 4. RESULTS AND DISCUSSION

**4.1. Predictive Ability.** The classification results for LVQ1+3 are presented in Table 4. Classification of both the training set and test set was good in all cases. Normally, the classification of samples in the test set would be expected to be much lower than for samples in the training set as they have not been used to model the data. Increasing the number of codebook vectors per class generally improved the overall prediction for the training set about as expected. Test set classification also improved as the number of codebook vectors was increased. There is good agreement between the %CC for the training set and that for the test set, which suggests that the model is reasonable. The increase in %CC for the test set requires some explanation and is probably because, in this study, the classes are not overlapping, but the boundaries are quite complex, so it is possible to draw a series of lines between each class, and there will be no samples on the borderline. Each grade forms a discrete subgroup, and the bootstrap and testing were performed using samples distributed over all grades, resulting in excellent test set prediction, even though there is a small chance of a grade not being represented in the training set. If a different strategy of removing grades rather than samples for validation were performed, this high classification ability of the test set would be unlikely to hold. However, for the current data set, because the number of grades is quite limited for each class, and because one of the aims is to model a polymer group over all known grades, the strategy of removing individual grades as test or bootstrap samples is not practicable.

The modal number of PCs selected for classification by the Mahalanobis distance was 4, and the average percent correctly classified using four PCs was 96.2%, which is comparable to using four codebook vectors per class for LVQ1+3.

When one codebook is used, prediction is relatively low probably because the model is not complex enough to represent the classes properly. There is good agreement between the %CC of the test and training sets when three codebook vectors are used per class, suggesting that this might be a good number of codebook vectors to use if the number of codebook vectors in each class is equal. For larger numbers of codebook vectors, the improvement in classification is small (1 or 2%).

**4.2. Classification.** During classification, some groups will inevitably be modeled better than others. It is useful to analyze which groups are misclassified more frequently and which grades are confused with each other.

For LVQ1+3, the two amorphous polymer groups, PS and ABS, showed the highest levels of misclassification and were mostly confused with each other, especially when a smaller number of codebook vectors were used. In general, however, all of the classes have fairly good classification rates (Table 4). An exception to this is the PBT samples when only one codebook is used per class, which has quite a poor prediction rate (65%) for this class. Increasing the number of codebook vectors drastically improves the prediction ability for this class, which suggests that using one codebook provides a much too simple model. Since the increase from three to five codebook vectors per class did not provide a significant amount of improvement in the classification rate, only three codebook vectors per class were used in further discussion below.

Table 5 shows the number of times a polymer grade was classified into a particular polymer using three codebook vectors per class. It can be seen that misclassifications mostly occur for different grades of a polymer class, rather than the polymer class as a whole, particularly for PS and ABS. PS is often blended with ABS to improve certain properties of the polymer.[41] Both R850E and SR(L)550 were known to be blends, whereas grades N2560 and 338 were considered to be “pure” PS, so it was expected that there may be some overlap between the PS and ABS groups. For increasing numbers of codebook vectors, however, SR(L)550 was classified with higher accuracy. This may be due to codebook vectors defining a border around localized regions that would normally be considered a different class, as more codebook vectors allow for more complex boundaries. The “pure” PS grades, N2560 and 338, showed low misclassification, even for ABS. The ABS grade JG (whose exact composition is not known as it is a grade of recycled polymers) probably has lower accuracy because it is being misclassified with PS samples of R850E.

Another poorly classified grade is the PET Eastman grade. Misclassified mostly as PBT, which is reasonable as PET and PBT have similar thermal properties, it is sometimes misclassified in several of the other polymer groups. This grade exhibits a particularly sharp peak in its $E''$ profile,

**Table 4.** Average Classification Results for All Classes Using LVQ1+3 and the Mahalanobis Distance

| | number of codebook vectors per class (total number) | group | | | | | | | | | average | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | PS | ABS | PCarb | LDPE | PP | HDPE | PA6 | PBT | PET | training set | test set |
| LVQ1+3 | 1 (9) | 79% | 86% | 100% | 88% | 89% | 92% | 95% | 89% | 66% | 86% | 85% |
| | 2 (18) | 88% | 89% | 100% | 92% | 94% | 97% | 97% | 96% | 82% | 92% | 91% |
| | 3 (27) | 89% | 92% | 100% | 95% | 96% | 99% | 98% | 99% | 90% | 95% | 94% |
| | 4 (36) | 95% | 92% | 100% | 97% | 98% | 99% | 98% | 98% | 94% | 97% | 96% |
| | 5 (45) | 94% | 94% | 100% | 96% | 98% | 100% | 98% | 100% | 97% | 98% | 97% |
| | 6 (54) | 97% | 94% | 100% | 97% | 99% | 99% | 98% | 100% | 98% | 98% | 98% |
| Mahalanobis distance (4 PCs) | | 97% | 96% | 81% | 99% | 100% | 100% | 97% | 62% | 100% | 99% | 96% |

**Table 5.** Classification of Individual Grades by LVQ1+3 Using Three Codebook Vectors Per Class

| | | polymer classification by LVQ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| polymer group | polymer grade | PS | ABS | PCarb | LDPE | PP | HDPE | PA6 | PBT | PET |
| PS | SR(L)550 | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | R850E | 65% | 35% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | N2560 | 99% | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | 338 | 94% | 2% | 0% | 0% | 0% | 0% | 3% | 0% | 2% |
| ABS | F332 | 0% | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | B432/E | 0% | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | JG | 40% | 60% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | Terluran GP-35 Natural | 0% | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| PCarb | Calibre 200 | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% | 0% |
| LDPE | FA3223 | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| | FB4230 | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| | 742gr | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| | Metler | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| | Shrivatsa s1 | 0% | 0% | 0% | 71% | 18% | 0% | 11% | 0% | 0% |
| | Shrivatsa s2 | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| PP | HA507MO | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% |
| | RD204CF | 0% | 0% | 0% | 1% | 99% | 0% | 0% | 0% | 0% |
| | HL512FB | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% |
| | BA212E | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% |
| | KSR4525 | 0% | 0% | 0% | 18% | 82% | 0% | 0% | 0% | 0% |
| HDPE | FS1470 | 0% | 0% | 0% | 0% | 0% | 96% | 0% | 3% | 0% |
| | B5823 | 0% | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% |
| | Padmex 60200 | 0% | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% |
| PA6 | Akulon K123 batch 2 | 0% | 0% | 0% | 0% | 0% | 0% | 97% | 3% | 0% |
| | Akulon K123 batch 1 | 0% | 0% | 0% | 0% | 0% | 0% | 99% | 1% | 0% |
| PBT | Arnite T04 200 | 0% | 0% | 0% | 0% | 0% | 1% | 1% | 99% | 0% |
| PET | Wellman | 2% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 98% |
| | Eastman | 1% | 0% | 0% | 2% | 2% | 0% | 6% | 23% | 67% |
| | 5143 | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 100% |
| | 5154 | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 3% | 97% |

**Table 6.** Classification of Polymer Groups by Mahalanobis Distance (Four PCs)

| actual class | predicted class | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | PS | ABS | PCarb | LDPE | PP | HDPE | PA6 | PBT | PET |
| PS | 97% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 3% |
| ABS | 4% | 96% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| PCarb | 1% | 0% | 81% | 0% | 0% | 0% | 0% | 0% | 18% |
| LDPE | 0% | 0% | 0% | 99% | 0% | 0% | 0% | 0% | 1% |
| PP | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% |
| HDPE | 0% | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% |
| PA6 | 0% | 0% | 0% | 0% | 0% | 0% | 97% | 0% | 3% |
| PBT | 0% | 2% | 0% | 0% | 0% | 5% | 2% | 62% | 29% |
| PET | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 100% |

**Table 7.** Classification of Polymer Groups by LVQ1+3 (Three Codebook Vectors)

| actual class | predicted class | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | PS | ABS | PCarb | LDPE | PP | HDPE | PA6 | PBT | PET |
| PS | 89% | 10% | 0% | 0% | 0% | 0% | 1% | 0% | 1% |
| ABS | 8% | 92% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| PCarb | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% | 0% |
| LDPE | 0% | 0% | 0% | 95% | 3% | 0% | 2% | 0% | 0% |
| PP | 0% | 0% | 0% | 4% | 96% | 0% | 0% | 0% | 0% |
| HDPE | 0% | 0% | 0% | 0% | 0% | 99% | 0% | 1% | 0% |
| PA6 | 0% | 0% | 0% | 0% | 0% | 0% | 98% | 2% | 0% |
| PBT | 0% | 0% | 0% | 0% | 0% | 1% | 1% | 99% | 0% |
| PET | 1% | 0% | 0% | 1% | 1% | 0% | 2% | 7% | 90% |

whereas for other grades of PET (and other polymers) this peak is much broader, causing the Eastman samples to be easily distinguishable from other samples of the PET polymer group.

The classification of samples and grades differs significantly when the Mahalanobis distance is used for classification. Tables 6 and 7 show the confusion matrices for classification by the Mahalanobis distance and by LVQ, respectively. In general, the polymer groups are classified very well by the Mahalanobis distance and, in some cases, better than for LVQ1+3. Some classes, however, have a much lower level of accuracy (e.g., PCarb). The overlap between PS and ABS is again evident, but to a lesser extent than for LVQ1+3. Table 8 shows the classification of individual grades by Mahalanobis distance. As for LVQ1+3, it appears to be a small number of grades that are misclassified more than others. In most cases, these are the same as for LVQ1+3 (JG, Shrivatsa s1).

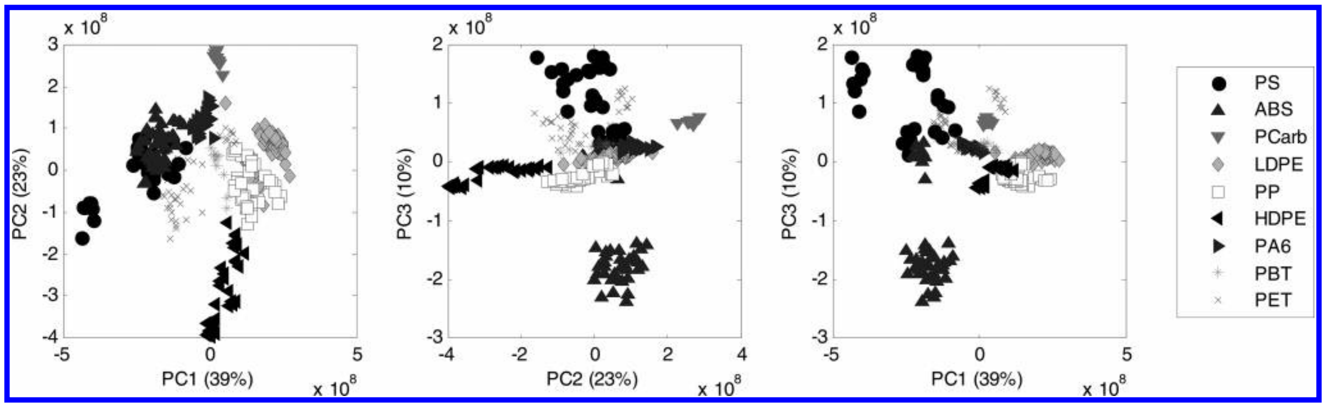


**Figure 8.** PCA scores plots of mean centered data.

**Table 8.** Classification of Individual Grades by Mahalanobis Distance Using Four PCs

| | | polymer classification by Mahalanobis distance | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| polymer group | polymer grade | PS | ABS | PCarb | LDPE | PP | HDPE | PA6 | PBT | PET |
| PS | SR(L)550 | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | R850E | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | N2560 | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | 338 | 90% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 10% |
| ABS | F332 | 0% | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | B432/E | 0% | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | JG | 22% | 78% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | Terluran GP-35 Natural | 0% | 100% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| PCarb | Calibre 200 | 1% | 0% | 81% | 0% | 0% | 0% | 0% | 0% | 18% |
| LDPE | FA3223 | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| | FB4230 | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| | 742gr | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| | Metler | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| | Shrivatsa s1 | 1% | 0% | 0% | 92% | 0% | 0% | 0% | 0% | 6% |
| | Shrivatsa s2 | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% | 0% |
| PP | HA507MO | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% |
| | RD204CF | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% |
| | HL512FB | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% |
| | BA212E | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% |
| | KSR4525 | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% | 0% |
| HDPE | FS1470 | 0% | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% |
| | B5823 | 0% | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% |
| | Padmex 60200 | 0% | 0% | 0% | 0% | 0% | 100% | 0% | 0% | 0% |
| PA6 | Akulon K123 batch 2 | 0% | 0% | 0% | 0% | 0% | 0% | 95% | 0% | 5% |
| | Akulon K123 batch 1 | 0% | 0% | 0% | 0% | 0% | 0% | 99% | 0% | 0% |
| PBT | Arnite T04 200 | 0% | 2% | 0% | 0% | 0% | 5% | 2% | 62% | 29% |
| PET | Wellman | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 100% |
| | Eastman | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 99% |
| | 5143 | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 100% |
| | 5154 | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 100% |

The scores plots from PCA (Figure 8) on the entire data set reveal where the samples fall in relation to one another. The overlap between PS and ABS is clearly visible and appears to be due to two grades that overlap (Figure 8c). The PET grades appear to have split into two separate clusters, the smaller group consisting of only one grade (Eastman), though the Mahalanobis distance had less difficulty classifying this grade correctly than LVQ. This may be because LVQ requires a codebook vector to be in both regions for correct classification, a scenario that is more likely as the number of codebook vectors increases.

Figure 8 also shows why PCarb may not be so well-classified by the Mahalanobis distance, as it appears to be similar in shape and location to the Eastman PET group, which is indeed where misclassifications are occurring. Figure 8c in particular exemplifies one of the advantages of using LVQ over the Mahalanobis distance for classifying PCarb polymers. The misclassifications by Mahalanobis distance arise because the grades of PET have formed two individual clusters. The Mahalanobis distance tries to model the entire class using an ellipsoid, which does not account for these individual clusters in the group data. LVQ is able to overcome this problem by placing codebook vectors within the individual clusters of a group, allowing much more complex borders to be modeled without making assumptions about the distribution of the class. The most poorly classified group using the Mahalanobis distance measure is PBT, which when considering the scores plots is unsurprising as it appears centrally between several of the polymer groups.

## 5. CONCLUSIONS

LVQ is an attractive approach for multiclass problems in chemistry; many classifiers such as PLS-DA or SVMs were originally developed for two class problems, and although they can be extended to multiclass decision making, it can sometimes be awkward. Approaches such as kNN and the

Mahalanobis distance are more suited where there are several classes but suffer limitations. In kNN, there are numerous difficulties with validation and differential sizes and spread of the classes. To use the Mahalanobis distance, classes need to be optimally distributed as ellipsoids, and such an approach cannot easily cope with quite complex boundaries. However, LVQ can be employed to find boundaries around classes; the more codebook vectors there are, the more complex these boundaries. In this paper, we study a mutliclass problem where the data are largely separable but complex boundaries (in the form of a series of hyperplanes determined by the codebook vectors) must be found. Under these circumstances, LVQ is an effective alternative not as yet employed routinely in chemical data analysis, and we demonstrate its effectiveness in a complex multiclass problem of polymers which achieves very high classification rates. Misclassifications can be analyzed using a confusion matrix and interpreted according to the grade, some of which would be expected to be more readily confused with others.

## 6. NOTATION

$I$, number of samples; $J$, number of measurements; $\theta$, temperature; $\mathbf{X}$, $I \times J$ data matrix; $A$, number of classes; $p_a$, number of codebook vectors for class $a$; $B$, total number of codebook vectors; $\mathbf{C}$, $B \times J$ matrix of codebook vectors; $T$, number of iterations; $t$, $t$th iteration; $r$, a randomly selected integer in the interval 1 to $I$; $\mathbf{x}_r$, $r$th $1 \times J$ sample vector of $\mathbf{X}$; $\alpha_t$, $t$th learning rate; $d_{rb}$, Euclidean distance between sample vector $\mathbf{x}_r$ and codebook vector $\mathbf{c}_b$; $k$, $k$th nearest neighbor; $d_{rk}$, Euclidean distance of sample $\mathbf{x}_r$ from its $k$th nearest codebook vector; $s$, window width; $W_t$, $t$th index of the winning codebook vector in $\mathbf{C}$; $V_t$, $t$th index of the second winning codebook vector in $\mathbf{C}$; $\mathbf{c}_w$, winning codebook vector where $w = W_t$; the nearest codebook neighbor of $\mathbf{x}_r$; $\mathbf{c}_v$, second winning codebook vector where $v = V_t$; the second nearest codebook neighbor of $\mathbf{x}_r$; $\mathbf{c}_u$, set of winning codebook vectors where $u \in \{v,w\}$; $\epsilon$, additional learning factor for LVQ3.

## REFERENCES AND NOTES

(1) Duda, R. O.; Hart, P. E.; Stork, D. G. *Pattern Classification*; 2nd ed.; Wiley: New York, 2001.
(2) Barker, M.; Rayens, W. Partial Least Squares for Discrimination. *J. Chemom.* **2003**, *17*, 166−173.
(3) Dixon, S. J.; Xu, Y.; Brereton, R. G.; Soini, H. A.; Novotny, M. V.; Oberzaucher, E.; Grammer, K.; Penn, D. J. Pattern Recognition of Gas Chromatography Mass Spectrometry of Human Volatiles in Sweat to Distinguish the Sex of Subjects and Determine Potential Discriminatory Marker Peaks. *Chemom. Intell. Lab. Syst.* in press. DOI: 10.1016/j. chemolab.2006.12.004.
(4) Vong, R.; Geladi, P.; Wold, S.; Esbensen, K. Source Contributions to Ambient Aerosol Calculated by Discriminant Partial Least Squares Regression (PLS). *J. Chemom.* **1988**, *2*, 281−296.
(5) Sjöström, M.; Wold, S.; Söderström, B. *Proceedings of PARC in Practice*; Elsevier Science Publishers: Amsterdam, 1985
(6) Ståhle, L.; Wold, S. Partial Least Squares Analysis with Cross-Validation for the Two-Class Problem: A Monte Carlo Study. *J. Chemom.* **1987**, *1*, 185−196.
(7) De Maesschalck, R.; Jouan-Rimbaud, D.; Massart, D. L. The Mahalanobis Distance. *Chemom. Intell. Lab. Syst.* **2000**, *50*, 1−18.
(8) Massart, D. L.; Vandeginste, B. G. M.; Buydens, L.; de Jong, S.; Lewi, P. J.; Smeyers-Verbeke, J. *Handbook of Chemometrics and Qualimetrics: Part B*; Elsevier: Amsterdam, 1998.
(9) Brereton, R. G. *Chemometrics: Data Analysis for the Laboratory and Chemical Plant*; John Wiley & Sons: Chichester, U. K., 2003.
(10) Zhuang, X.; Huang, Y.; Palaniappan, K.; Zhao, Y. Gaussian Mixture Density Modeling, Decomposition, and Applications. *IEEE Trans. Image Process.* **1996**, *5*, 1293−1302.
(11) Specht, D. F. In *Probabilistic Neural Networks for Classification, Mapping, or Associative Memory*, IEEE International Conference on Neural Networks, San Diego, CA, Jul 24−27, 1988; IEEE: Piscataway, NJ, 1988; pp 525−532.
(12) Zomer, S.; Brereton, R. G.; Carter, J. F.; Eckers, C. Support Vector Machines for the Discrimination of Analytical Chemical Data: Application to the Determination of Tablet Production by Pyrolysis-Gas Chromatography-Mass Spectrometry. *Analyst* **2004**, *129*, 175−181.
(13) Zomer, S.; Sánchez, M. D. N.; Brereton, R. G.; Pavón, J. L. P. Active Learning Support Vector Machines for Optimal Sample Selection in Classification. *J. Chemom.* **2004**, *18*, 294−305.
(14) Belousov, A. I.; Verzakov, S. A.; von Frese, J. A Flexible Classification Approach With Optimal Generalisation Performance: Support Vector Machines. *Chemom. Intell. Lab. Syst.* **2002**, *64*, 15−25.
(15) Vapnik, V. N.; Lerner, A. Pattern Recognition Using Generalised Portrait Method. *Autom. Remote Control (Engl. Trans.)* **1963**, *24*, 774−780.
(16) Vapnik, V. N. *Statistical Learning Theory*; Wiley: New York, 1998.
(17) Yu, X.; Zomer, S.; Brereton, R. G. Support Vector Machines: A Recent Method for Classification in Chemometrics. *Crit. Rev. Anal. Chem.* **2006**, *36*, 177−188.
(18) Bredensteiner, E. J.; Bennett, K. P. Multicategory Classification by Support Vector Machines. *Comput. Optim. Appl.* **1999**, *12*, 53−79.
(19) Hsu, C.-W.; Lin, C.-J. A Comparison of Methods for Multiclass Support Vector Machines. *IEEE Trans. Neural Networks* **2002**, *13*, 415−425.
(20) Aiolli, F.; Sperduti, A. Multiclass Classification with Multi-Prototype Support Vector Machines. *J. Mach. Learn. Res.* **2005**, *6*, 817−850.
(21) Kohonen, T., The Self-Organizing Map. *Proc. IEEE* **1990**, *78*, 1464−1480.
(22) Kohonen, T. *Self-Organizing Maps*; Springer-Verlag: Berlin, 1997.
(23) Kohonen, T.; Kangas, J.; Laaksonen, J.; Torkkola, K. In *LVQPAK: A Software Package for the Correct Application of Learning Vector Quantization Algorithms*; IJCNN, International Joint Conference on Neural Networks, Baltimore, Maryland, June 7−11, 1992; International Neural Network Society: New York, 1992; pp 725−730.
(24) Efron, B.; Tibshirani, R. J. *An Introduction to The Bootstrap*; Chapman and Hall: New York, 1993.
(25) Brereton, R. G. Consequences of Sample Size, Variable Selection, and Model Validation and Optimisation, for Predicting Classification Ability From Analytical Data. *TrAC, Trends Anal. Chem.* **2006**, *25*, 1103−1111.
(26) Hjorth, J. S. U. *Computer Intensive Statistical Methods: Validation, Model Selection and Bootstrap*; Chapman and Hall: New York, 1994.
(27) Faria, R.; Duncan, J. C.; Brereton, R. G. Dynamic Mechanical Analysis and Chemometrics for Polymer Identification. *Polym. Test.* **2007**, *26*, 402−412.
(28) Lukasiak, B. M.; Faria, R.; Zomer, S.; Brereton, R. G.; Duncan, J. C. Pattern Recognition for the Analysis of Polymeric Materials. *Analyst* **2006**, *131*, 73−80.
(29) Lukasiak, B. M.; Zomer, S.; Brereton, R. G.; Faria, R.; Duncan, J. C. Pattern Recognition and Feature Selection for the Discrimination Between Grades of Commercial Plastics. *Chemom. Intell. Lab. Syst.* **2007**, *87*, 18−25.
(30) Read, B. E.; Dean, G. D.; Duncan, J. C. *Determination of Dynamic Moduli and Loss Factors*; John Wiley & Sons: New York, 1991.
(31) Sperling, L. H. *Introduction to Physical Polymer Science*; Wiley: New York, 2006.
(32) Nicholson, J. W. *The Chemistry of Polymers*; Royal Society of Chemistry: Cambridge, U. K., 2006.
(33) Vesanto, J.; Alhoniemi, E. Clustering of the Self-Organizing Map. *IEEE Trans. Neural Networks* **2000**, *11*, 586−600.
(34) Erwin, E.; Obermayer, K.; Schulten, K. Self-Organizing Maps: Ordering, Convergence Properties and Energy Functions. *Biol. Cybern.* **1992**, *V67*, 47−55.
(35) Okabe, A.; Boots, B.; Sugihara, K. *Spacial Tesselations. Concepts and Applications of Voronoi Diagrams*, 2nd ed.; John Wiley & Sons: Chichester, U. K., 2000.
(36) De Stefano, C.; D'Elia, G.; Marcelli, A. In *A Dynamic Approach to Learning Vector Quantization*; Proceedings of the 17th International Conference on Pattern Recognition, Cambridge U. K, Aug. 23−26, 2004; IEEE Computer Society Press: Los Alamitos, CA, 2004; pp 601−604.
(37) Cagnoni, S.; Valli, G. In *OSLVQ: A Training Strategy for Optimum-Size Learning Vector Quantization Classifiers*; IEEE Inter-

national Conference on Neural Networks, Orlando, Florida, Jun. 27−29, 1994; IEEE Neural Networks Council: Piscataway, NJ, 1994; pp 762−765.
(38) Kohonen, T. In *Improved Versions of Learning Vector Quantization*; 1990 IJCNN International Joint Conference on Neural Networks, Washington, DC, Jan. 15−19, 1990; Lawrence Erlbaum Associates: Hillsdale, NJ, 1990; pp 545−550.
(39) Wold, S. Cross-Validatory Estimation of the Number of Components in Factor and Principal Components Models. *Technometrics* **1978**, *20*, 397−405
(40) Kohavi, R. In *A Study of Cross-Validation of Bootstrap for Accuracy Estimation and Model Selection*; International Joint Conference on Artificial Intelligence, Montreal, Quebec, Canada, Aug. 20−25, 1995; Morgan Kaufmann Publishers: Burlington, MA, 1995; pp 1137−1143
(41) Roberto Greco, A. S. Polycarbonate/ABS Blends: A Literature Review. *Adv. Polym. Technol.* **1994**, *13*, 249−258.

CI700019Q