upclass: An R Package for Updating Model-Based Classification Rules

Niamh Russell
University College Dublin

Laura Cribbin
University of Limerick

Thomas Brendan Murphy
University College Dublin

Abstract

Standard methods for classification use labeled data to establish criteria for assigning unlabeled data to groups. However, the unlabeled data which need to be classified often contain important information about the structure of the groups, despite the group membership of these observations being unknown. A new R package called **upclass** is presented which uses both labeled and unlabeled data to construct a model-based classification method. The method uses the EM algorithm to obtain maximum likelihood estimates of the model parameters and classifications for the unlabeled data. It can be shown to perform better than classical methods, particularly in cases where few observations are labeled.

Keywords: classification, EM algorithm, **mclust**, R, semi-supervised classification.

1. Introduction

Classification techniques are employed regularly in a wide variety of application areas. Examples include food science applications where studies are carried out to establish whether products are correctly labeled (e.g., Caetano, Üstün, Hennessy, Smeyers-Verbeke, Melssen, Downey, Buydens, and Heyden 2007; Toher, Downey, and Murphy 2007, 2011); botanical investigations to identify rare plants (e.g., Pouteau, Meyer, Taputuarai, and Stoll 2012) and medical diagnostic applications to identify whether patients have a particular disease or condition (e.g., Fan, Murphy, Byrne, Brennan, Fitzpatrick, and Watson 2011). It is important to devise effective rules in order to reduce potential errors.

Classification methods require a labeled dataset so that the number of groups and the structure of groups can be inferred. The task is to classify any unlabeled observations into the correct groups. Traditionally, a classification rule is developed using the fully labeled data which can then be used to classify any new unlabeled data as it becomes available. Extensive reviews of classification methods include Ripley (1996) and McLachlan (1992).

Semi-supervised classification methods use both the labeled *and* unlabeled data to develop a classifier for the unlabeled observations. These methods exploit the idea that even though the group memberships of the unlabeled data are unknown, these data carry important information about the group parameters (e.g., McLachlan 1977; O'Neill 1978; Dean, Murphy, and Downey 2006; Chapelle, Schölkopf, and Zien 2006). These methods provide a framework for updating a classification rule using unlabeled observations, so that more accurate classifications can be obtained. A number of semi-supervised classification methods have been

developed including model-based methods (e.g., Dean *et al.* 2006; McNicholas 2010; Murphy, Dean, and Raftery 2010; Toher *et al.* 2011) and machine learning methods (e.g., Joachims 1999; Wang, Chen, and Zhou 2012). Detailed reviews of semi-supervised classification include Chapelle *et al.* (2006) and Zhu and Goldberg (2009).

Herein, we present the R package **upclass** which implements the (semi-supervised) updated model-based classification method as developed in Dean *et al.* (2006). This method starts by estimating the unknown labels using a standard model-based classification method and then combines them with the labeled observations to form the complete-data. The EM algorithm is utilized where the parameters and estimated unknown labels are iteratively updated until convergence. This yields maximum likelihood estimates for the parameters in the model and estimates group membership for the unlabeled observations. The algorithm for standard model-based classification is outlined in Section 2 and the algorithm for the updated version is described in more detail in Section 3.

In Section 3.2, using the well known olive oil data set (Forina, Armanino, Lanteri, and Tiscornia 1983) as an example, we give a short comparison of the effectiveness of the updated method versus a classical method for a case where only a small proportion of the data is labeled.

In Section 4, we will illustrate the use of each function in the **upclass** package by working through examples using the olive oil data.

The R package implementing the methodology described in this article is available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=upclass`.

2. Model-based methods

Discriminant analysis is concerned with classifying data into predefined groups while clustering sets out to cluster data into a previously undefined number of groups. In this section, we will outline the model-based approach to clustering (Section 2.1) and discriminant analysis (Section 2.2). The updated classification method which will be developed in Section 3 uses a hybrid of the statistical models underlying model-based discriminant analysis and clustering.

2.1. Model-based clustering

Model-based clustering (Banfield and Raftery 1993; Fraley and Raftery 2002, 2007) as implemented in the **mclust** package (Fraley, Raftery, Murphy, and Scrucca 2012) is used for clustering data into groups, where the number of groups G is unknown.

Model-based clustering is formulated as follows, we assume that there are G clusters, where each cluster arises with probability τ_g and data within each cluster follows a normal distribution with cluster specific mean μ_g and covariance $\mathbf{\Sigma}_g$. That is, the data are characterized by a finite mixture of normal distributions.

Hence, the density of each observation can be given by,

$$f(y) = \sum_{g=1}^{G} \tau_g f(y|\mu_g, \mathbf{\Sigma}_g),$$

where $f(\cdot)$ is a multivariate normal density.

It is worth noting that the assumption of multivariate normal distributed clusters implies that the clusters are elliptical in shape. Banfield and Raftery (1993) proposed that constraints are placed on the covariance matrices in such a way as to produce different shapes and sizes for the clusters. A modified eigenvalue decomposition of $\mathbf{\Sigma}_g$ is used to implement these variations. This decomposition can be written as

$$\mathbf{\Sigma}_g = \lambda_g \mathbf{D}_g \mathbf{A}_g \mathbf{D}_g^T,$$

where λ_g – is a constant which controls the cluster volume

$\mathbf{D}_g$ – is an orthogonal matrix of eigenvectors which control the orientation/direction of the clusters

$\mathbf{A}_g$ – is a diagonal matrix, with entries proportional to the eigenvalues, which control the shape of the cluster.

We can restrict each part of the covariance $\mathbf{\Sigma}_g$ in different ways, resulting in fourteen different possible models (Biernacki, Celeux, Govaert, and Langrognet 2006). Throughout this paper, we will consider the ten covariance structures implemented in **mclust** (Fraley *et al.* 2012), as displayed in Figure 1 and Table 1. Each letter in the name of a model corresponds to the constraint placed on the volume, shape and orientation respectively. The constraint can be equal (E), variable (V) or identity (I). Consider, for example, the EEV model for the covariance. If data are fitted by this model, then each cluster has the same volume and the same shape but the orientation of each cluster is allowed to differ.

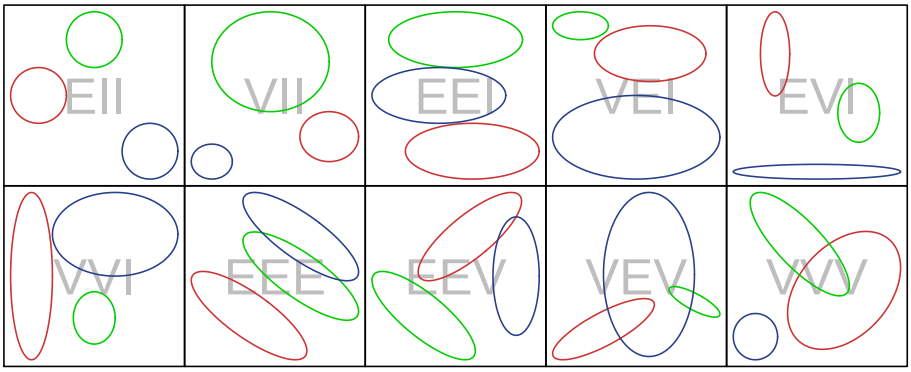


Figure 1: Examples of clusters under each covariance restriction.

As shown in Figure 1, the various covariance restrictions result in a different combination of cluster shapes in each model. The constraints yield parsimonious models which facilitate a more flexible modeling strategy beyond assuming unequal covariance (VVV) or equal covariance (EEE).

The model parameters are estimated using maximum likelihood via the EM algorithm (Dempster, Laird, and Rubin 1977). The EM algorithm, a technique used to find maximum likelihood estimates in cases where there are missing data, is made up of two steps, the Expectation (E)

Table 1: Covariance decompositions available.

Model	Volume	Shape	Orientation	Covariance $\mathbf{\Sigma}_g$
EII	Equal	Spherical		$\lambda\mathbf{I}$
VII	Variable	Spherical		$\lambda_g\mathbf{I}$
EEI	Equal	Equal	Axis aligned	$\lambda\mathbf{A}$
VEI	Variable	Equal	Axis aligned	$\lambda_g\mathbf{A}$
EVI	Equal	Variable	Axis aligned	$\lambda\mathbf{A}_g$
VVI	Variable	Variable	Axis aligned	$\lambda_g\mathbf{A}_g$
EEE	Equal	Equal	Equal	$\lambda\mathbf{D}\mathbf{A}\mathbf{D}^T$
EEV	Equal	Equal	Variable	$\lambda\mathbf{D}_g\mathbf{A}\mathbf{D}_g^T$
VEV	Variable	Equal	Variable	$\lambda_g\mathbf{D}_g\mathbf{A}\mathbf{D}_g^T$
VVV	Variable	Variable	Variable	$\lambda_g\mathbf{D}_g\mathbf{A}_g\mathbf{D}_g^T$

and the Maximization (M) steps. Since the data are from a mixture model,

$$f(y) = \sum_{g=1}^{G} \tau_g f(y|\mu_g, \mathbf{\Sigma}_g),$$

we can write the likelihood as the product over this density, evaluated at each $\mathbf{y}_m$,

$$L(\tau, \mu, \mathbf{\Sigma}|\mathbf{y}_M) = \prod_{m=1}^{M} \left[\sum_{g=1}^{G} \tau_g f(y_m|\mu_g, \mathbf{\Sigma}_g) \right], \tag{1}$$

and the log-likelihood as

$$l(\tau, \mu, \mathbf{\Sigma}|\mathbf{y}_M) = \sum_{m=1}^{M} \log \left[\sum_{g=1}^{G} \tau_g f(y_m|\mu_g, \mathbf{\Sigma}_g) \right],$$

where $\tau = (\tau_1, \tau_2, \ldots, \tau_G)$, $\mu = (\mu_1, \mu_2, \ldots, \mu_G)$ and $\mathbf{\Sigma} = (\mathbf{\Sigma}_1, \mathbf{\Sigma}_2, \ldots, \mathbf{\Sigma}_G)$. It is difficult to maximize the log-likelihood directly. To resolve this, we introduce indicator variables, z_{mg}, which represent the unknown labels of each observation,

$$\text{where } \; z_{mg} = \begin{cases} 1 & \text{if } y_m \text{ is from group } g \\ 0 & \text{otherwise.} \end{cases}$$

The complete-data likelihood can now be written as

$$L(\tau, \mu, \mathbf{\Sigma}|\mathbf{y}_M, \mathbf{z}_M) = \prod_{m=1}^{M} \left[\tau_g g(y_m|\mu_g, \mathbf{\Sigma}_g)\right]^{z_{mg}}, \tag{2}$$

and the complete-data log-likelihood is of the form

$$l(\tau, \mu, \mathbf{\Sigma}|\mathbf{y}_M, \mathbf{z}_M) = \sum_{m=1}^{M} \sum_{g=1}^{G} z_{mg} \left[\log \tau_g + \log f(y_m|\mu_g, \mathbf{\Sigma}_g)\right]. \tag{3}$$

The E-step of the algorithm replaces the z_{mg} values in (3) with their conditional expected values (5), thus yielding the expected complete-data log-likelihood. The M-step of the algorithm then maximizes the expected complete-data log-likelihood function. The algorithm is iterated until convergence and the final z_{mg} values provide the posterior probability that observation m belongs to group g. Each observation is classified to the group with maximum *a posteriori* (MAP) probability.

Further details for the algorithm are provided in Section 3 where the EM algorithm for the updated method of model-based classification is described.

2.2. Model-based discriminant analysis

Two classical classification methods are linear (LDA) and quadratic (QDA) discriminant analysis. Both methods can be seen as model-based discriminant analysis methods based on a similar model to that outlined in Section 2.1 but where the group membership for each observation is known. When implementing LDA, the covariance matrix is assumed equal for each group, which corresponds to the EEE covariance structure. Whereas in QDA, each group is allowed to have its own unconstrained covariance matrix, which corresponds to the VVV covariance structure.

In the context of discriminant analysis there are two types of data: labeled data for which the group memberships are known and unlabeled data where they are unknown. Discriminant analysis uses labeled data to estimate model parameters which are used to create a classification rule. This rule can then be used to classify the unlabeled data.

Let $(\mathbf{x}_N, \mathbf{l}_N)$ be the labeled data, where the observations are denoted by $\mathbf{x}_N = (x_1, x_2, \ldots, x_N)$ and their labels by $\mathbf{l}_N = (l_1, l_2, \ldots, l_N)$. The unlabeled data will be represented by $\mathbf{y}_M$ where $\mathbf{y}_M = (y_1, y_2, \ldots, y_M)$ and the unknown labels are $\mathbf{z}_M = (z_1, z_2, \ldots, z_M)$. The likelihood of the labeled data can be written as:

$$L(\tau, \mu, \mathbf{\Sigma}|\mathbf{x}_N, \mathbf{l}_N) = \prod_{n=1}^{N}\prod_{g=1}^{G}[\tau_g f(x_n|\mu_g, \mathbf{\Sigma}_g)]^{l_{ng}}. \tag{4}$$

Hence, the log-likelihood is

$$l(\tau, \mu, \mathbf{\Sigma}|\mathbf{x}_N, \mathbf{l}_N) = \sum_{n=1}^{N}\sum_{g=1}^{G} l_{ng}[\log \tau_g + \log f(x_n|\mu_g, \mathbf{\Sigma}_g)].$$

$$\text{where } \; l_{ng} = \begin{cases} 1 & \text{if } x_n \text{ belongs to group } g \\ 0 & \text{otherwise.} \end{cases}$$

The function $l(\tau, \mu, \mathbf{\Sigma}|\mathbf{x}_N, \mathbf{l}_N)$ can be maximized with respect to $(\tau_g, \mu_g, \mathbf{\Sigma}_g)$ to obtain maximum likelihood estimates for the parameters $(\hat{\tau}_g, \hat{\mu}_g, \hat{\mathbf{\Sigma}}_g)$ in the model. Using these estimates, calculated from the labeled data, the expected value of the unknown labels $\mathbf{z}_M$ can be computed as

$$\hat{z}_{mg} = \frac{\hat{\tau}_g f(y_m|\hat{\mu}_g, \hat{\mathbf{\Sigma}}_g)}{\sum_{g=1}^{G} \hat{\tau}_{g'} f(y_m|\hat{\mu}_{g'}, \hat{\mathbf{\Sigma}}_{g'})}. \tag{5}$$

These *a posteriori* probabilities of group membership for each observation can be used to derive the maximum *a posteriori* (MAP) predicted group membership for each observation.

2.3. Model selection

In implementing model-based discriminant analysis and clustering techniques, the model of the data must be known. If the model is not known, it is recommended to fit every model to the data and calculate the Bayesian Information Criterion (BIC) value (Schwarz 1978; Kass and Raftery 1995) for each model. The BIC value is calculated in such a way where it penalizes for a large number of parameters and rewards for a large likelihood value.

$$\text{BIC} = 2\log(L) - k\log(n)$$

where L – is the likelihood of the data
k – is the number of model parameters
n – is the number of observations.

The model with the highest BIC value is selected. While this does not always guarantee the lowest misclassification rate, in practice it often selects a close to optimal model (Biernacki and Govaert 1999). Biernacki and Govaert (1999) provide an overview of other model selection criteria for model-based clustering and classification.

3. Updating method

The backbone of the method employed by **upclass** is the idea that the unlabeled data may potentially contain important information about the overall data even though their group memberships are unknown (Dean *et al.* 2006). This information can help give a clearer picture of the structure of the groups in the data. Earlier work in using labeled data to update model-based classification rules has been carried out by McLachlan (1975, 1977), Ganesalingam and McLachlan (1978) and O'Neill (1978) amongst others. More recent work includes Nigam, McCallum, and Mitchell (2006), McNicholas (2010) and Murphy *et al.* (2010).

We have observed $(\mathbf{x}_N, \mathbf{l}_N, \mathbf{y}_M)$ and unknown $\mathbf{z}_M$. So the observed likelihood is of the form

$$L(\tau, \mu, \mathbf{\Sigma}|\mathbf{x}_N, \mathbf{l}_N, \mathbf{y}_M) = \underbrace{\prod_{n=1}^{N}\prod_{g=1}^{G}\left[\tau_g f(x_n|\mu_g, \mathbf{\Sigma}_g)\right]^{l_{ng}}}_{\text{labeled data}} \underbrace{\prod_{m=1}^{M}\left[\sum_{g=1}^{G}\tau_g f(y_m|\mu_g, \mathbf{\Sigma}_g)\right]}_{\text{unlabeled data}}; \quad (6)$$

this is the product of the likelihood for model-based discriminant analysis (4) and model-based clustering (1). If we treat the unknown labels as missing data, we can write the likelihood for the complete-data as

$$L_c(\tau, \mu, \mathbf{\Sigma}|\mathbf{x}_N, \mathbf{l}_N, \mathbf{y}_M, \mathbf{z}_M) = \underbrace{\prod_{n=1}^{N}\prod_{g=1}^{G}\left[\tau_g f(x_n|\mu_g, \mathbf{\Sigma}_g)\right]^{l_{ng}}}_{\text{labeled data}} \underbrace{\prod_{m=1}^{M}\prod_{g=1}^{G}\left[\tau_g f(y_m|\mu_g, \mathbf{\Sigma}_g)\right]^{z_{mg}}}_{\text{unlabeled data}}; \quad (7)$$

this is a product of the likelihood for model-based discriminant analysis (4) and the complete-data likelihood for model-based clustering (2). The package maximizes the likelihood (6) using the EM algorithm, which utilizes the complete-data likelihood (7), to find maximum likelihood estimates for the unknown parameters and hence estimates for the unknown labels.

3.1. How it works

There are four general steps used to implement this updated classification rule. They iterate through the EM algorithm and are made up of the following:

Step 1 Let $k = 0$. Find initial values for the parameter estimates in the model. Only the labeled data $(\mathbf{x}_N, \mathbf{l}_N)$ are used here along with the M-step of the EM algorithm. This is equivalent to performing classical model-based discriminant analysis to obtain starting values for the model parameters.

Step 2 Using the current parameter estimates, $\tau^{(k)}$, $\mu^{(k)}$ and $\mathbf{\Sigma}^{(k)}$, calculate the expected value of the unknown labels through the E-step,

$$\hat{z}_{mg}^{(k+1)} = \frac{\hat{\tau}_g^{(k)} f(y_m|\hat{\mu}_g^{(k)}, \mathbf{\Sigma}_g^{(k)})}{\sum_{g'=1}^{G} \hat{\tau}_{g'}^{(k)} f(y_m|\hat{\mu}_{g'}^{(k)}, \mathbf{\Sigma}_{g'}^{(k)})}.$$

Step 3 Combine $(\mathbf{x}_N, \mathbf{l}_N)$ and $(\mathbf{y}_M, \hat{\mathbf{z}}_M^{(k+1)})$ to form the complete-data. Using the complete-data, calculate new parameter estimates for the model, $\tau^{(k+1)}$, $\mu^{(k+1)}$ and $\mathbf{\Sigma}^{(k+1)}$, through the M-step by maximizing the log complete-data likelihood (7).

Step 4 Check for convergence using the Aitken acceleration convergence criterion, by default. There is a simpler convergence option also, see Section 4.5. If convergence has been reached, stop. If not, set $k = k + 1$ and return to Step 2 where new estimates for the unknown labels are calculated followed by new parameter estimates.

The parameter estimates used in Step 3 are of the following form. The estimate of $\hat{\tau}_g^{(k+1)}$ can be seen as the average number of observations in each group,

$$\hat{\tau}_g^{(k+1)} = \frac{\sum_{n=1}^{N} l_{ng} + \sum_{m=1}^{M} \hat{z}_{mg}^{(k+1)}}{N + M},$$

and $\hat{\mu}_g^{(k+1)}$ is a weighted average of the observations, where the labels and their estimates are used as weights,

$$\hat{\mu}_g^{(k+1)} = \frac{\sum_{n=1}^{N} l_{ng}\mathbf{x}_n + \sum_{m=1}^{M} \hat{z}_{mg}^{(k+1)}\mathbf{y}_m}{\sum_{n=1}^{N} l_{ng} + \sum_{m=1}^{M} \hat{z}_{mg}^{(k+1)}}.$$

The estimation of $\mathbf{\Sigma}_g$ depends on the constraints placed on the covariance matrix. For example, if the model was VVV, the estimate for $\hat{\mathbf{\Sigma}}_g^{(k+1)}$ would look like;

$$\hat{\mathbf{\Sigma}}_g^{(k+1)} = \frac{\sum_{n=1}^{N} l_{ng}(\mathbf{x}_n - \hat{\mu}_g^{(k+1)})(\mathbf{x}_n - \hat{\mu}_g^{(k+1)}) + \sum_{m=1}^{M} \hat{z}_{mg}^{(k)}(\mathbf{y}_m - \hat{\mu}_g^{(k+1)})(\mathbf{y}_m - \hat{\mu}_g^{(k+1)})'}{\sum_{n=1}^{N} l_{ng} + \sum_{m=1}^{M} \hat{z}_{mg}^{(k)}}.$$

For further details on parameter estimation and how the covariance matrix for each model can be calculated, see Bensmail and Celeux (1996). Once the final converged estimates of the model have been obtained, these maximize the observed-data likelihood (6). The resulting parameters and $\hat{z}_{mg}$ values form the updated classification rule.

3.2. Comparing the two methods

We compare how the two methods perform using the olive oil data set (Forina *et al.* 1983) as an example. This data set is made up of eight variables which measure the percentage composition of eight fatty acids on 572 olive oils. Each oil is from one of three Italian regions; North Italy, South Italy and Sardinia. Note that we did not classify by geographical area within region, neither did we include this finer geographical area variable in our training/test data.

The data were randomly split into training (labeled) and test (unlabeled) data with a given percentage being assigned as training data. This process was repeated 200 times and classical model-based discriminant analysis and the updated method were fit to each split of the data. In each case, the best model was selected using BIC and the classification performance was recorded for this model.

Table 2 shows the results of classification tests using model-based discriminant analysis and the updated method, and also the number of times each model was selected at each level. It can be seen that VVV is selected as the best model until only 15% of the data are labeled. The VVV model, as the one with the most parameters, requires the training data to be of a certain size to work efficiently. On some data splits where fewer than 15% of the observations are labeled, some groups had insufficient observations to fit the VVV model so simpler models were selected; in these cases the selected models enumerated by the numbers in brackets.

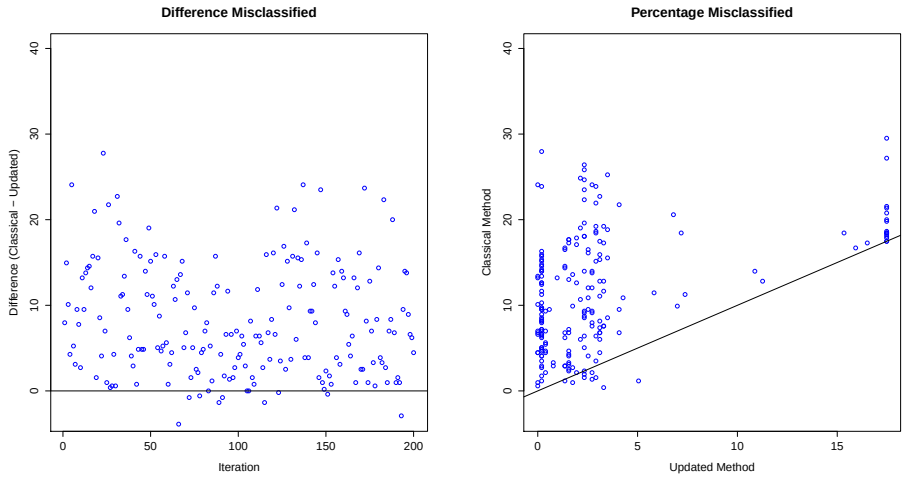
Table 2: Olive oil data: The proportion of labeled data versus the average misclassification rate (reported as a percentage) for the classical and updated methods. The frequency that each covariance structure was selected is also shown.

Labeled data	Classical method	Models		Updated method	Models	
90%	0.08	VVV		0.04	VVV	
80%	0.10	VVV		0.07	VVV	
70%	0.15	VVV		0.08	VVV	
60%	0.19	VVV		0.09	VVV	
50%	0.32	VVV		0.12	VVV	
40%	0.45	VVV		0.16	VVV	
30%	0.79	VVV		0.25	VVV	
20%	2.24	VVV		0.57	VVV	
15%	5.04	VVV	(195)	1.29	VVV	(194)
		VEV	(2)		VEV	(6)
		EEE	(3)			
10%	11.46	VVV	(141)	3.30	VVV	(137)
		VEV	(42)		VEV	(59)
		EEE	(17)		EEV	(10)

The results in Table 2 show the updated method outperforming the classical method at *every* level, although the methods show comparable results when more than 30% of the data are labeled. However, at the 30% level and lower, the results from the updated method far surpass those from the classical method.

At the 10% level, the difference between the two methods is very apparent. In this case, 515 observations are unlabeled out of the total of 572. The classical method misclassified nearly 11.5% of the unlabeled observations, which corresponds to 59 observations. The updated method has misclassified only 3.3% of the unlabeled data, which corresponds to only 17 observations.

Figure 2 compares the results of the two methods for each of the 200 iterations at the 10% level of labeled data.



(a) (b)

Figure 2: Olive oil data: Comparing the classical and updated methods at the 10% level: (a) shows the difference in the number of misclassified observations for the classical and updated methods. (b) shows a scatter plot of the number of misclassified observations for the classical and updated methods.

Figure 2(a) shows that the updated method almost always gives better results than the classical method. The updated method yielded a lower misclassification rate on 188 iterations out of 200, the same misclassification rate on 3 iterations and an inferior misclassification rate on 9 iteration. Figure 2(b) shows the number of observations that were misclassified by each method on each iteration. On 24 iterations, the classical method misclassified at least 20% of the unlabeled observations reaching a maximum of misclassifying almost 30% of the unlabeled observations for one iteration.

4. The software

In this section we will discuss how to use the package **upclass** in R (R Development Core Team 2011). It can be implemented in a supervised or semi-supervised mode. It makes extensive use of the package **mclust** (Fraley *et al.* 2012). If **mclust** is not installed, **upclass** will install it.

The package is available on CRAN and can be installed using the following code.

```
R> install.packages("upclass")
R> library(upclass)
```

4.1. Setting up the data

To illustrate the use of the functions in **upclass**, we will again employ the olive oil dataset (Forina *et al.* 1983). The dataset is found in the **classifly** package developed by Wickham (2011). We will set up the data in the following way.

```
R> data(olives)
R> X<-as.matrix(olives[,-c(1:2)])
R> cl<-as.matrix(olives[,2])
R> N<-dim(X)[1]
R> indtrain<-sort(sample(1:N, N * 0.2))
R> Xtrain<-X[indtrain,]
R> cltrain<-cl[indtrain]
R> indtest<-setdiff(1:N, indtrain)
R> Xtest<-X[indtest,]
R> cltest<-cl[indtest]
```

This randomly assigns 20% of the data (114 observations) as labeled data (`Xtrain`, `cltrain`), where `Xtrain` are the observations and `cltrain` are their labels. The remaining data are unlabeled. The observations are stored as `Xtest`, and their removed labels as `cltest`.

4.2. Supervised model

We first illustrate the basic use of the classical model fit, with the default model EEE in mind. We could change the model to any of the ten available in **mclust** (Table 1).

```
R> fitnoup <- noupclassifymodel(Xtrain, cltrain, Xtest)
```

Fitting this model is equivalent to classifying the data using LDA. Since we have the correct labels to hand, we can include them in our model as follows; this will result in the function reporting classification performance results.

```
R> fitnoup1 <- noupclassifymodel(Xtrain, cltrain, Xtest, cltest)
```

We can thus extract a table comparing the maximum *a posteriori* labels with the true ones, the Brier score (Brier 1950), the BIC criterion (Schwarz 1978), as well as a reported number of misclassified observations, as can be seen in the sample output included next.

```
Labels for test data provided

Total Misclassified:
 12

Brier Score:
```

```
 8.74595

BIC:
 -9054.496

Classification Table
    1  2   3
1 255  1   1
2   0 76   0
3   0 10 115

----------------------------------------
```

However, it is likely that we would not have a particular model in mind when starting out our analysis. In this case we can use the `noupclassify` command, as follows.

```
R> fitnoupall <- noupclassify(Xtrain, cltrain, Xtest, cltest)
```

This runs `noupclassifymodel` for a selected list of models or for all of them (the default is to fit all ten models in Table 1); see `modelvec` function later in Section 4.4.

The `noupclassify` function returns the 'best' model as well as the associated output for this model. As outlined in Section 2.3, the best model is considered to be the one with the largest BIC value.

A subset of the output for the same data sample is provided below. As can be seen, it suggests that the VVV model is the best for these data. This is in keeping with our findings in Section 3.2.

```
$Best

Function Call:
noupclassifymodel(Xtrain = Xtrain, cltrain = cltrain, Xtest = Xtest,
    cltest = cltest, modelName = modelName, reportrate = reportrate)

No in Training Set:
 114

No in Test Set:
 458

No of Groups:
 3

Model Name:
 VVV

----------------------------------------
```

```
----------------------------------------

Labels for test data provided

Total Misclassified:
 4

Brier Score:
 2.949363

BIC:
 -8833.488

Classification Table
    1  2   3
1 257  0   0
2   2 74   0
3   1  1 123

----------------------------------------
```

The output for all the other models that were fitted is also stored in the output from `noupclassify`.

Note that the BIC criterion usually selects the model with the lowest misclassification rate, but not always. With this data split, the VEI model only had one misclassification but had a less satisfactory BIC, as can be seen from the code below.

```
R> fitnoupall$VEI

Model Name:
 VEI

----------------------------------------

----------------------------------------

Labels for test data provided

Total Misclassified:
 1

Brier Score:
 1.015307

BIC:
 -9660.548
```

```
Classification Table
    1  2   3
1 256  0   1
2   0 76   0
3   0  0 125

----------------------------------------
```

4.3. Semi-supervised model

The real power of this package lies in the semi-supervised model. The functions work in a similar fashion. For those who wish to fit a specific model, for example the VEV model, the code below will do this. The default model, as before, is EEE.

```
R> fitup <- upclassifymodel(Xtrain, cltrain, Xtest,cltest, modelName="VEV")
```

If the labels are not available, the `cltest` vector can be omitted. A sample of the output follows.

```
Labels for test data provided

Total Misclassified:
 8

Brier Score:
 7.986492

BIC:
 -43507.49

Classification Table
    1  2   3
1 257  0   0
2   0 76   0
3   7  1 117

----------------------------------------
```

If you have selected a model and wish to try the classification for a range of models, the `upclassify` command can be used. Here, we use the same data split and the full range of models. It can be seen that VVV is selected again, but with only one misclassification this time.

```
Model Name:
 VVV
```

```
----------------------------------------

----------------------------------------

Labels for test data provided

Total Misclassified:
 1

Brier Score:
 0.9995319

BIC:
 -42739.35

Classification Table
    1  2   3
1 257  0   0
2   0 76   0
3   0  1 124

----------------------------------------
```

Both `upclassifymodel` and `noupclassifymodel` have output with the user-defined class of `upclassfit`. A `print()` and a `summary()` for this class of output has been developed. The `print` has already been detailed but the output from `summary` is more extensive.

```
R> print(fitup)
```

```
R> summary(fitup)
```

Note that in both functions, the available components of the output list are itemized so particular results can be picked out.

```
Available Components:
 [1] "call"       "Ntrain"     "Ntest"      "d"          "G"          "iter"
 [7] "converged"  "ModelName"  "parameters" "reportrate" "train"      "test"
[13] "ll"         "bic"
```

There is also a `plot()` function which produces a simple heat map of the classification matrix. An example appears in Figure 3.

```
R> plot(fitup)
```

4.4. Models

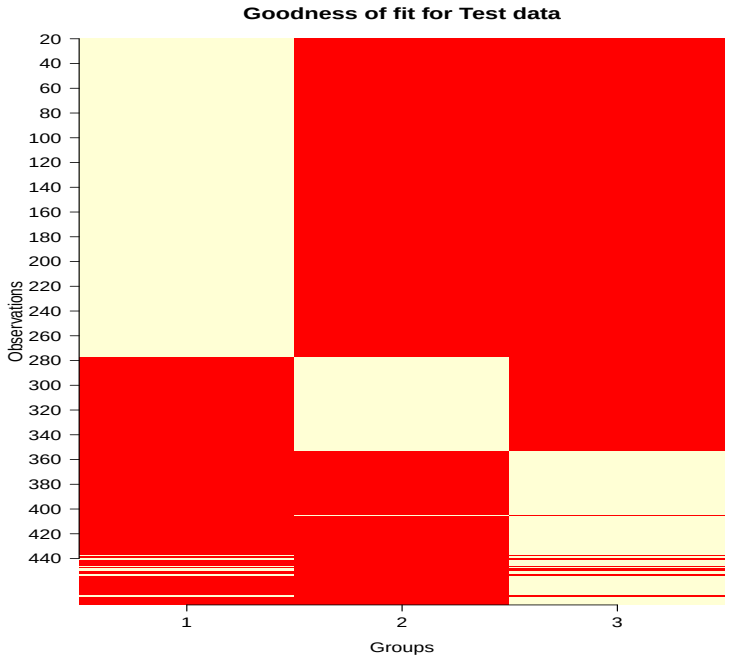


Figure 3: Heat map of the classification of the olive oil data using the VEV model.

`modelvec` is a function which outputs all the available models for univariate/multivariate normally distributed data. It can be used alongside `modelscope` from some of the previous functions.

There is one input argument; the dimension of the data. If the data are one-dimensional, or univariate, there are only two possible models here.

```
R> modelvec(1)

[1] "E" "V"
```

If the data are multivariate, all ten available models are returned as here where there are four variables in the data set.

```
R> modelvec(4)

 [1] "EII" "VII" "EEI" "VEI" "EVI" "VVI" "EEE" "EEV"
 [9] "VEV" "VVV"
```

If only a few of these models are needed, they can be extracted and stored in a vector, i.e., `scope`. Perhaps while running `noupclassify` or `upclassify`, we wish only to test models with equal volume. These can be entered manually as a vector string through the `modelscope` argument, or else `modelvec` can be used as in the following:

```
R> scope <- modelvec(4)[c(1, 3, 5, 7, 8)]
R> upclassify(Xtrain, cltrain, Xtest, modelscope = scope)
```

So here, `upclassifymodel` is run only over models with equal volume (see Fraley and Raftery 2002).

4.5. Convergence

Both `upclassify` and `upclassifymodel` in **upclass** can make use of the function `Aitken` by setting the argument `Aitken` to `TRUE`. This function takes in a vector of three consecutive log-likelihoods and estimates the final converged maximized log-likelihood using the method of Aitken acceleration described by Böhning, Dietz, Schaub, Shlattmann, and Lindsay (1994). The calling functions can then decide if the log-likelihood has converged based on some specified tolerance, defaulted to 10^{-5} in the case of the **upclass** functions.

This function could, of course, be used by itself, as in the following snippet.

```
R> ll<-c(-261, -257.46,-256.4)
R> Aitken(ll)

$ll
[1] -256.4

$linf
[1] -254.8869

$a
[1] 0.299435
```

`ll` gives the current estimate for the log-likelihood, while `linf` gives the estimate of the converged value and if the difference between the two should be less than some specified tolerance, convergence can be said to have been reached.

A simpler convergence criterion can be used by setting `Aitken` to `FALSE`. This method achieves convergence when two successive values of `ll` have a difference smaller than `tol`. The simpler convergence criterion has been shown to be less strict than the Aitken one (McNicholas, Murphy, McDaid, and Frost 2010).

5. Discussion

We have presented a problem in data classification that occurs frequently in many areas, namely to classify unlabeled observations when only in possession of a small amount of labeled data. As such it would be of interest to researchers in food science, medical diagnostics, botany, or any area where such a scenario is commonplace.

We have introduced a R package called **upclass** which goes some way to addressing this problem. As we saw in Section 3 we can take advantage of the complete-data (both labeled and unlabeled) to create a classifier. The package is an implementation of the method developed in Dean *et al.* (2006), and takes advantage of the EM algorithm functionality developed by Fraley *et al.* (2012) in the package **mclust**.

We have described the functions available in the package in Section 4. The user can use the function `upclassify` to classify his data over the full range of models described in Section 2.1,

or to select one or more suitable models. It is possible to vary the parameters to control convergence criteria, and also control the output produced. The function `noupclassify` is provided to carry out supervised classification, if desired. Functions are provided to interrogate the output from the functions in the package.

We have shown (in Table 2) that this method can provide significantly better results at low levels of labeling, than supervised classification.

The main limitation of the idea is that we assume that each group can be modeled by a normal distribution as we discussed in Section 2.1; in some applications this assumption may not be appropriate. Also, at very low levels of labeling, **upclass** cannot fit all possible models for the data, and must choose among the models with a suitably reduced number of parameters where model fitting is feasible.

The package is currently based on the ten covariance structures in **mclust** however fourteen covariance structures are possible within the modified eigen-decomposition. It would be interesting to extend to the approach to all fourteen covariance structures (Biernacki *et al.* 2006) to increase the flexibility of this approach further.

A possible avenue for exploration, is to cater for the situation where not all groups are represented in the training set. This becomes more and more likely at very low levels of labeling, and might have other applications if any new datapoint belongs to a previously unknown group.

Acknowledgements

This research is supported by the Programme for Research In Third Level Institutions (PRTLI) Cycle 5 and co-funded by the European Regional Development Fund, and the Science Foundation Ireland Research Frontiers Programme (2007/RFP/MATF281).

References

Banfield JD, Raftery AE (1993). "Model-Based Gaussian and Non-Gaussian Clustering." *Biometrics. Journal of the Biometric Society*, **49**(3), 803–821.

Bensmail H, Celeux G (1996). "Regularized Gaussian Discriminant Analysis through Eigenvalue Decomposition." *Journal of the American Statistical Association*, **91**(436), 1743–1748.

Biernacki C, Celeux G, Govaert G, Langrognet F (2006). "Model-Based Cluster and Discriminant Analysis with the MIXMOD Software." *Computational Statistics & Data Analysis*, **51**(2), 587–600.

Biernacki C, Govaert G (1999). "Choosing Models in Model-Based Clustering and Discriminant Analysis." *Journal of Statistical Computation and Simulation*, **64**, 49–71.

Böhning D, Dietz E, Schaub R, Shlattmann P, Lindsay B (1994). "The Distribution of the Likelihood Ratio for Mixtures of Densities from the One-Parameter Exponential Family." *Annals of the Institute of Statistical Mathematics*, **46**(2), 373–388.

Brier GW (1950). "Verification of Forecasts Expressed in Terms of Probability." *Monthly Weather Review*, **78**(1), 1–3.

Caetano S, Üstün B, Hennessy S, Smeyers-Verbeke J, Melssen W, Downey G, Buydens L, Heyden YV (2007). "Geographical Classification of Olive Oils by the Application of CART and SVM to their FT-IR." *Journal of Chemometrics*, **21**(7-9), 324–334.

Chapelle O, Schölkopf B, Zien A (eds.) (2006). *Semi-Supervised Learning*. MIT Press, Cambridge, MA.

Dean N, Murphy TB, Downey G (2006). "Using Unlabelled Data to Update Classification Rules with Applications in Food Authenticity Studies." *Journal of the Royal Statistical Society. Series C. Applied Statistics*, **55**(1), 1–14.

Dempster AP, Laird NM, Rubin DB (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society. Series B. Methodological*, **39**(1), 1–38. With Discussion.

Fan Y, Murphy TB, Byrne J, Brennan L, Fitzpatrick J, Watson RWG (2011). "Applying Random Forests to Identify Biomarker Panels in Serum 2D-DIGE Data for the Detection and Staging of Prostate Cancer." *Journal of Proteome Research*, **10**(3), 1361–1373.

Forina M, Armanino C, Lanteri S, Tiscornia E (1983). "Classification of Olive Oils from Their Fatty Acid Composition." In H Martens, H Russwurm Jr (eds.), *Food Research and Data Analysis*, pp. 189–214. Applied Science Publishers, London.

Fraley C, Raftery A (2007). "Model-Based Methods of Classification: Using the mclust Software in Chemometrics." *Journal of Statistical Software*, **18**(6), 1–13.

Fraley C, Raftery AE (2002). "Model-Based Clustering, Discriminant Analysis, and Density Estimation." *Journal of the American Statistical Association*, **97**(458), 611–631.

Fraley C, Raftery AE, Murphy TB, Scrucca L (2012). "mclust Version 4 for R: Normal Mixture Modeling for Model-Based Clustering, Classification, and Density Estimation." *Technical Report 597*, Department of Statistics, University of Washington.

Ganesalingam S, McLachlan GJ (1978). "The Efficiency of a Linear Discriminant Function Based on Unclassified Initial Samples." *Biometrika*, **65**(3), 658–662.

Joachims T (1999). "Transductive Inference for Text Classification using Support Vector Machines." In *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 200–209. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. ISBN 1-55860-612-2.

Kass R, Raftery AE (1995). "Bayes Factors and Model Uncertainty." *Journal of the American Statistical Association*, **90**, 773–795.

McLachlan G (1992). *Discriminant Analysis and Statistical Pattern Recognition*. Wiley, New York.

McLachlan GJ (1975). "Iterative Reclassification Procedure for Constructing an Asymptotically Optimal Rule of Allocation in Discriminant Analysis." *Journal of the American Statistical Association*, **70**, 365–369.

McLachlan GJ (1977). "Estimating the Linear Discriminant Function from Initial Samples Containing a Small Number of Unclassified Observations." *Journal of the American Statistical Association*, **72**(358), 403–406.

McNicholas PD (2010). "Model-Based Classification using Latent Gaussian Mixture Models." *Journal of Statistical Planning and Inference*, **140**(5), 1175–1181.

McNicholas PD, Murphy TB, McDaid AF, Frost D (2010). "Serial and Parallel Implementations of Model-Based Clustering via Parsimonious Gaussian Mixture Models." *Computational Statistics & Data Analysis*, **54**(3), 711–723.

Murphy TB, Dean N, Raftery AE (2010). "Variable Selection and Updating in Model-Based Discriminant Analysis for High Dimensional Data with Food Authenticity Applications." *Annals of Applied Statistics*, **4**(1), 396–421.

Nigam K, McCallum A, Mitchell T (2006). "Semi-Supervised Text Classification Using EM." In O Chapelle, B Schölkopf, A Zien (eds.), *Semi-Supervised Learning*. MIT Press, Boston.

O'Neill TJ (1978). "Normal Discrimination with Unclassified Observations." *Journal of the American Statistical Association*, **73**(364), 821–826.

Pouteau R, Meyer JY, Taputuarai R, Stoll B (2012). "Support Vector Machines to Map Rare and Endangered Native Plants in Pacific Islands Forests." *Ecological Informatics*, **9**, 37–46.

R Development Core Team (2011). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Ripley BD (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press.

Schwarz G (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**(2), 461–464.

Toher D, Downey G, Murphy TB (2007). "A Comparison of Model-Based and Regression Classification Techniques Applied to Near Infrared Spectroscopic Data in Food Authentication Studies." *Chemometrics and Intelligent Laboratory Systems*, **89**(2), 102–115.

Toher D, Downey G, Murphy TB (2011). "Semi-Supervised Linear Discriminant Analysis." *Journal of Chemometrics*, **25**(12), 621–630.

Wang Y, Chen S, Zhou ZH (2012). "New Semi-Supervised Classification Method Based on Modified Cluster Assumption." *IEEE Transactions on Neural Networks and Learning Systems*, **23**(5), 689–702.

Wickham H (2011). *classifly: Explore Classification Models in High Dimensions*. R package version 0.3, URL http://CRAN.R-project.org/package=classifly.

Zhu X, Goldberg AB (2009). "Introduction to Semi-Supervised Learning." *Synthesis Lectures on Artificial Intelligence and Machine Learning*, **3**(1), 1–130.

Affiliation:

Niamh Russell
Complex and Adaptive Systems Laboratory
& School of Mathematical Sciences
Belfield Office Park
Clonskeagh
Dublin 4.
E-mail: niamh.russell.1@ucdconnect.ie