Improving Risk Management

RIESGOS-CM

Análisis, Gestión y Aplicaciones

P2009/ESP-1685

Technical Report 2011.1

Improving Fraud Detection Modeling

Javier Cano Cancela, David Ríos Insua and Raul Moreno Izquierdo

http://www.analisisderiesgos.org

Improving Fraud Detection Modeling

Javier Cano[1], David Ríos Insua[2], Raúl Moreno[3]

[1]Department of Statistics and Operations Research, Rey Juan Carlos University
[2]Royal Academy of Sciences, Spain
[3]Rey Juan Carlos University

Abstract

This document describes improvements upon previous fraud detection approaches. We emphasize Bayesian mixture modeling with an unknown number of mixture components for classification and outlier detection, together with the need of properly evaluating economically the consequences of fraud detection. We sketch how to apply this approach to CDR data.

1 Introduction

Fraud detection, a continuously evolving discipline, involves identifying fraud as quickly as possible. The common fraudster adapts his strategies to commit new fraud when it becomes known that one detection method is in place. This makes anti-fraud a very dynamic and adversarial game: the development of new fraud detection methods is made more difficult by the fact that the exchange of ideas in fraud detection is severely limited, as revealing such information would aid fraudsters in evading detection.

Previous approaches to fraud detection are based on methods like Support Vector Machines, influence diagrams and (non model based) cluster analysis. These models may be used for classification, when data can be labeled as fraudulent or not, or outlier detection, when labels are not available, in the framework of fraud detection. Typically, tools for fraud detection will attempt to sort a transaction or event as being legal or fraudulent, by generating a score and comparing it against a threshold. The efficiency of such methods is not outstanding, but they could be a temporal solution. But fraudsters learn fast, or sometimes there is no well-defined profile, or attacks are very rare. Taken together, all these facts mean that data mining based systems will not uncover any new fraudster behavior until they are accurate for new behaviors, and that even very accurate systems will be so flooded with false alarms so as to become useless.

Any of these methods finds some percentage of real attacks, but also provides some percentage of false alarms, and it seems necessary to evaluate whether the benefits of finding and stopping those attacks outweigh the costs, in money,

company image, etc. associated with false alarms. However, in a sense, statistically based approaches to fraud detection tend to focus on minimizing the number of false positives and negatives detected, without paying much attention to the evaluation of such errors or using a very simplistic cost function. From our point of view, forecasting such costs is important as it impacts on the utility function of the anti-fraud decision maker, and usually decision makers are not well prepared to deal with uncertainty measures associated with predictions.

We shall describe here a number of improvements upon standard approaches. The underlying theme is Bayesian mixture modeling with an unknown number of components for five key reasons:

- They provide realistic and flexible models in many contexts.
- There are efficient computational schemes for inference and prediction with such models.
- Classification and outlier detection may be naturally described within such models.
- There is a natural way to infer the number of components in the mixture.
- Prior information may be coherently incorporated within such framework.

We start by introducing basic concepts in mixture modeling and then illustrate four mixture models that might be relevant in fraud detection, as described:

- Modeling with normal mixtures.
- Modeling with exponential mixtures.
- Modeling with gamma mixtures.
- Hierarchical modeling with normal mixtures.

We provide inference and prediction schemes for such models.

Next, we describe how to perform classification and outlier detection with such models for fraud detection purposes. We do it in three stages: classification for the case in which labels are available; outlier detection for the case in which labels are not available; and a combined approach as labels become available on the spot, which we designate online fraud detection. We then pay attention to the need of properly considering costs associated with fraud detection. Finally, we sketch how the case of CDR's could be dealt with.

In what follows, except when noted, we consider only the univariate case, although the extension to multivariate distributions is straightforward.

2 Inference and prediction with mixture models

2.1 General framework

The basic mixture model for independent scalar or vector observations $\boldsymbol{x} = (x_1, \ldots, x_n)$ is expressed as

$$x_i \sim \sum_{j=1}^{M} w_j f(\cdot|\theta_j), \qquad \text{independently for } i = 1, 2, \cdots, n, \tag{1}$$

where $f(\cdot|\theta_j)$ is a given parametric family of densities indexed by a scalar or vector parameter $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_M)$, and the weights must hold $\sum_{j=1}^{M} w_j = 1$, $w_j \geq 0$. In such analysis, we postulate a heterogenous population consisting of groups $j = 1, \ldots, M$ of sizes proportional to w_j, from which the random sample is drawn. We aim at providing inference about the problem unknowns:

- the number of components, M;
- the component parameters, $\boldsymbol{\theta}$;
- and the mixture weights, $\boldsymbol{w} = (w_1, \ldots, w_M)$.

The mixture weights allow to define group labels (the identity of the group from which each observation is drawn), $\boldsymbol{z} = (z_1, \ldots, z_n)$, as

$$\Pr\{z_i = j|w, M\} = w_j, \tag{2}$$

for $i = 1, \ldots, n$; $j = 1, 2, \cdots, M$. Given the values of the z_i, the observations are drawn from their respective individual subpopulations

$$x_i|\theta, \boldsymbol{z} \sim f(\cdot|\theta_{z_i}), \qquad \text{independently for } i = 1, 2, \cdots, n. \tag{3}$$

In a Bayesian framework, the unknowns M, $\boldsymbol{w}$ and $\boldsymbol{\theta}$ are regarded as drawn from appropriate prior distributions. The joint distribution of all variables can be written, by appropriately imposing conditional independencies, as

$$p(M, \boldsymbol{w}, \boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{x}) = p(M)p(\boldsymbol{w}|M)p(\boldsymbol{z}|\boldsymbol{w}, M)p(\boldsymbol{\theta}|M)p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{z}).$$

Furthermore, we could allow the priors for M, $\boldsymbol{w}$ and $\boldsymbol{\theta}$ to depend on hyperparameters λ, $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$ respectively. These will be drawn from independent hyperpriors. The joint distribution of all variables is then expressed by the factorization

$$\begin{aligned} p(\lambda, \boldsymbol{\delta}, \boldsymbol{\eta}, M, \boldsymbol{w}, \boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{x}) = p(\lambda)p(\boldsymbol{\delta})p(\boldsymbol{\eta})\times \\ p(M|\lambda)p(\boldsymbol{w}|M, \boldsymbol{\delta})p(\boldsymbol{z}|\boldsymbol{w}, M)p(\boldsymbol{\theta}|M, \boldsymbol{\eta})p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{z}). \end{aligned} \tag{4}$$

The key computational tool will be the reversible jump MCMC, see Green (1995), which is a random sweep Metropolis-Hastings method, see French and Ríos Insua (2000), adapted for general state spaces. Other variable dimension

Monte Carlo samplers include jump diffusions Grenander and Miller (1994) and birth-death samplers (Stephens, 2000). For RJMCMC, letting $\boldsymbol{y}$ denote the state variable (in our case $\boldsymbol{y}$ is the complete set of unknowns $\boldsymbol{y} \equiv (M, \boldsymbol{w}, \boldsymbol{z}, \boldsymbol{\theta})$), and $\pi(\boldsymbol{y})$ the target probability measure (the posterior distribution), we consider a countable family of move types, indexed by $m = 1, \ldots$. When the current state is $\boldsymbol{y}$, a move type m and destination $\boldsymbol{y}'$ are proposed, and the move is accepted with probability

$$\alpha_m(\boldsymbol{y}, \boldsymbol{y}') = \min\left\{1, \frac{\pi(d\boldsymbol{y}')q_m(\boldsymbol{y}', d\boldsymbol{y})}{\pi(d\boldsymbol{y})q_m(\boldsymbol{y}, d\boldsymbol{y}')}\right\}, \tag{5}$$

where $q_m(\boldsymbol{y}', d\boldsymbol{y})$ is a probability measure.

For a move type that does not change the dimension of the parameter, (5) reduces to the usual Metropolis-Hastings acceptance probability, using an ordinary ratio of densities. For dimension-changing moves, e.g., from $\boldsymbol{y}$ to $\boldsymbol{y}'$ in a higher dimensional space, this will be very often implemented by drawing a vector of continuous random variables $\boldsymbol{u}$, independent of $\boldsymbol{y}$, and setting $\boldsymbol{y}'$ by using an invertible deterministic function $\boldsymbol{y}'(\boldsymbol{y}, \boldsymbol{u})$. The reverse move, from $\boldsymbol{y}'$ to $\boldsymbol{y}$ can be accomplished by using the inverse transformation, so that the proposal is deterministic. Then, (5) reduces to

$$\min\left\{1, \frac{p(\boldsymbol{y}'|\boldsymbol{x})r_m(\boldsymbol{y}')}{p(\boldsymbol{y}|\boldsymbol{x})r_m(\boldsymbol{y})q(\boldsymbol{u})}\left|\frac{\partial \boldsymbol{y}'}{\partial(\boldsymbol{y}, \boldsymbol{u})}\right|\right\}, \tag{6}$$

where $r_m(\boldsymbol{y})$ is the probability of choosing move type m when in state $\boldsymbol{y}$, and $q(\boldsymbol{u})$ is the density function of $\boldsymbol{u}$. The final term in the ratio is the Jacobian arising from the change of variable from $(\boldsymbol{y}, \boldsymbol{u})$ to $\boldsymbol{y}'$.

We shall describe now several potentially relevant mixtures in fraud detection and specify the corresponding sampling schemes.

2.2 Normal mixtures[1]

Normal mixtures will be relevant when transactions are described by vectors of continuous features. In the univariate normal case (easily generalizable to the multivariate case), the generic parameter $\boldsymbol{\theta}$ is the pair mean and variance $(\mu_j, \sigma_j^2),\ j = 1, \ldots, M$, and the general model (1) becomes

$$x_i \sim \sum_{j=1}^{M} w_j f_{\mathcal{N}}(x_i|\mu_j, \sigma_j^2),\ i = 1, \ldots, n,$$

where

$$f_{\mathcal{N}}(x_i|\mu_j, \sigma_j^2) = \frac{1}{\sigma_j\sqrt{2\pi}} \exp\left(-\frac{(x_i - \mu_j)^2}{2\sigma_j^2}\right).$$

[1]Based on Richardson and Green (1997), and French and Rios Insua (2000)

Our prior distribution assumptions will be that μ_j and σ_j^{-2} are all drawn independently, with normal and gamma priors

$$\mu_j \sim \mathcal{N}(\xi, \kappa^{-1}) \qquad \text{and} \qquad \sigma_j^{-2} \sim \mathcal{G}(\alpha, \beta),$$

so that the generic $\boldsymbol{\eta}$ becomes $\boldsymbol{\eta} = (\xi, \kappa, \alpha, \beta)$.

Concerning the labels of the components, z_i, we use, unless stated otherwise, the labeling in which the μ_j are in increasing numerical order; thus the joint prior distribution of the parameters is $M!$ times the product of the individual normal and gamma densities, restricted to the set $\mu_1 < \mu_2 < \cdots < \mu_M$. The prior on $\boldsymbol{w}$ will be a symmetric Dirichlet $\boldsymbol{w} \sim \mathcal{D}(\delta, \delta, \ldots, \delta)$. It is also necessary to adopt a proper prior distribution for M. The usual choice is a Poisson distribution with $M \sim \mathcal{P}(\lambda)$, but we shall use instead also a uniform distribution $M \sim \mathcal{U}(1, M_{\max})$.

This model allows us to specify the posterior conditionals given the other parameters as follows:

$$\boldsymbol{w}|\boldsymbol{x}, \boldsymbol{z} \sim \mathcal{D}(\delta + n_1, \ldots, \delta + n_M),$$

where $n_j = \#\{i : z_i = j\}$, for $j = 1, \ldots, M$ is the size of the j-th cluster and $\sum_{j=1}^{M} n_j = n$.

$$\mu_j|\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\sigma} \sim \mathcal{N}\left(\frac{\sigma_j^{-2}\sum_{i:z_i=j} x_i + \kappa\xi}{\sigma_j^{-2} n_j + \kappa}, (\sigma_j^{-2} n_j + \kappa)^{-1}\right),\ j = 1, \ldots, M$$

To preserve the ordering constraint on the $\{\mu_j\}$, the full conditional is used only to generate a proposal and is accepted provided that the ordering is unchanged.

$$\sigma_j^{-2}|\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\mu}, \beta \sim \mathcal{G}\Big(\alpha + \tfrac{1}{2} n_j, \beta + \tfrac{1}{2}\sum_{i:z_i=j} (x_i - \mu_j)^2\Big),\ j = 1, \ldots, M$$

$$\Pr\{z_i = j|x_i, \boldsymbol{w}, \boldsymbol{\mu}, \boldsymbol{\sigma}\} \propto \frac{w_j}{\sigma_j} \exp\left(-\frac{(x_i - \mu_j)^2}{2\sigma_j^2}\right),\ i = 1, \ldots, n,\ j = 1, \ldots, M \tag{7}$$

The only hyperparameter we are not treating as fixed, $\boldsymbol{\beta}$, has prior $\mathcal{G}(g, h)$, and posterior

$$\boldsymbol{\beta}|\boldsymbol{x}, \boldsymbol{\sigma} \sim \mathcal{G}\left(g + M\alpha, h + \sum_j \sigma_j^{-2}\right)$$

For all these variables we use Gibbs steps.

Then, if $\boldsymbol{y}^t = (\boldsymbol{w}^t, \boldsymbol{z}^t, \boldsymbol{\mu}^t, \boldsymbol{\sigma}^t, \boldsymbol{\beta}^t)$ designates the state vector at the t-th iteration, the sampler goes through the following steps:

Algorithm 2.1. RJ sampler for variable size normal mixtures

i. Start with arbitrary values $(\boldsymbol{w}^0, \boldsymbol{z}^0, \boldsymbol{\mu}^0, \boldsymbol{\sigma}^0, \boldsymbol{\beta}^0, M^0)$, $t = 0$.

ii. Until convergence, iterate through

1. Generate $\boldsymbol{w}^{t+1} \sim \boldsymbol{w}|\boldsymbol{x}, \boldsymbol{z}^t$.
2. Generate $\mu_j^{t+1} \sim \mu_j|x_j, \boldsymbol{\sigma}^t, \boldsymbol{z}^t, \xi, \kappa,\ j = 1, \ldots, M$.
3. Generate $(\sigma_j^{-2})^{t+1} \sim \sigma_j^{-2}|x_j, \boldsymbol{z}^t, \boldsymbol{\mu}^{t+1}, \alpha, \beta,\ j = 1, \ldots, M$.
4. Generate $\boldsymbol{z}_i^{t+1} \sim \boldsymbol{z}_i|x_j, \boldsymbol{w}^{t+1}, \boldsymbol{\mu}^{t+1}, \boldsymbol{\sigma}^{t+1},\ i = 1, \ldots, , n$.
5. Generate $\beta_j^{t+1} \sim \beta_j|x_j, \boldsymbol{\sigma}^{t+1}, g, M, \alpha, h,\ j = 1, \ldots, M$.
6. Split a mixture component into two, or vice versa.
7. Birth or death of an empty component.
8. Set $t = t + 1$.

For the split or combine move [6], the reversible jump mechanism is needed. Recall that we need to design these moves in tandem: they form a reversible pair. The strategy is to choose the proposal distribution according to informal considerations suggesting a reasonable probability of acceptance, but strictly subject to the requirement of dimension matching. Having done so, conformation with the detailed balance condition is determined by the acceptance probability (6).

In move [6], we make a random choice between attempting to split or combine, with probabilities b_M and $d_M = 1 - b_M$ respectively, depending on M. Of course, $d_1 = 0$ and $b_{M_{\max}} = 0$, where $M_{\max}$ is the maximum value allowed for M, and otherwise we choose $d_M = b_M = 0.5$, for $M = 2, 3, \ldots, M_{\max} - 1$. Our combine proposal begins by choosing a pair of components (j_1, j_2) at random, that are adjacent in terms of the current value of their means, i.e.

$$\mu_{j_1} < \mu_{j_2}, \qquad \text{with no other } \mu_j \text{ in the interval } [\mu_{j_1}, \mu_{j_2}] \tag{8}$$

These two components are merged, reducing M by 1. In doing so, forming a new component here labeled j^*, we have to reallocate all those observations x_i with $z_i = j_1$ or $z_i = j_2$ and create values for $(w_{j^*}, \mu_{j^*}, \sigma_{j^*})$. The reallocation is simply done by setting $z_i = j^*$, whereas the other parameters are assigned by matching the zeroth, first and second moments of the new component to those

of the combination of the two that it replaces:

$$\left.\begin{aligned} w_{j^*} &= w_{j_1} + w_{j_2} \\ w_{j^*}\mu_{j^*} &= w_{j_1}\mu_{j_1} + w_{j_2}\mu_{j_2} \\ w_{j^*}(\mu_{j^*}^2 + \sigma_{j^*}^2) &= w_{j_1}(\mu_{j_1}^2 + \sigma_{j_1}^2) + w_{j_2}(\mu_{j_2}^2 + \sigma_{j_2}^2) \end{aligned}\right\} \tag{9}$$

This combined proposal is deterministic once the discrete choices of j_1 and j_2 have been made.

The reverse split proposal is now largely determined. A component j^* is chosen at random and split into two, labeled j_1 and j_2, with weights and parameters conforming to equations (9). There are 3 degrees of freedom in achieving this, so we need to generate a three-dimensional random vector u to specify the new parameters. We use beta distributions

$$u_1 \sim \mathcal{B}e(2,2) \qquad u_2 \sim \mathcal{B}e(2,2) \qquad u_3 \sim \mathcal{B}e(1,1)$$

for this, and set

$$\begin{aligned} w_{j_1} &= w_{j^*}u_1 \qquad w_{j_2} = w_{j^*}(1-u_1), \\ \mu_{j_1} &= \mu_{j^*} - u_2\sigma_{j^*}\sqrt{\frac{w_{j_2}}{w_{j_1}}}, \\ \mu_{j_2} &= \mu_{j^*} + u_2\sigma_{j^*}\sqrt{\frac{w_{j_1}}{w_{j_2}}}, \\ \sigma_{j_1}^2 &= u_3(1-u_2^2)\sigma_{j^*}^2\frac{w_{j^*}}{w_{j_1}}, \\ \sigma_{j_2}^2 &= (1-u_3)(1-u_2^2)\sigma_{j^*}^2\frac{w_{j^*}}{w_{j_2}}, \end{aligned}$$

which provide all six required weights and parameters, satisfying equations (9). It can be readily shown that these are indeed valid, with weights and variances positive. At this point, we check whether the adjacency condition (8) is satisfied. If not, the move is rejected, as the (split, combine) pair could not then be reversible. If the test is passed, it remains only to propose the reallocation of those x_i with $z_i = j^*$ between j_1 and j_2. This is done analogously to the standard Gibbs allocation move; see equation (7).

The acceptance probabilities for the split or combine moves, calculated from expression (6), have quite a convoluted form. For the split move the probability

is $\min(1, A)$, where

$$\begin{aligned} A = (\text{likelihood ratio}) & \frac{p(M+1)}{p(M)}(M+1)\frac{w_{j_1}^{\delta-1+l_1} w_{j_2}^{\delta-1+l_2}}{w_{j^*}^{\delta-1+l_1+l_2} B(\delta, M\delta)} \\ & \times \sqrt{\frac{\kappa}{2\pi}} \exp[-\tfrac{1}{2}\kappa\{(\mu_{j_1}-\xi)^2 + (\mu_{j_2}-\xi)^2 - (\mu_{j^*}-\xi)^2\}] \\ & \times \frac{\beta^\alpha}{\Gamma(\alpha)} \left(\frac{\sigma_{j_1}^2 \sigma_{j_2}^2}{\sigma_{j^*}^2}\right)^{-\alpha-1} \exp\{-\beta(\sigma_{j_1}^{-2} + \sigma_{j_2}^{-2} - \sigma_{j^*}^{-2})\} \\ & \times \frac{d_{M+1}}{b_M P_{\text{alloc}}} \{g_{2,2}(u_1) g_{2,2}(u_2) g_{1,1}(u_3)\}^{-1} \\ & \times \frac{w_{j^*} |\mu_{j_1} - \mu_{j_2}| \sigma_{j_1}^2 \sigma_{j_2}^2}{u_2(1-u_2^2) u_3 (1-u_3) \sigma_{j^*}^2} \end{aligned} \tag{10}$$

where M is the number of components before the split, l_1 and l_2 are the numbers of observations proposed to be assigned to j_1 and j_2, $B(\cdot, \cdot)$ is the beta function, P_{alloc} is the probability that this particular allocation is made, $g_{p,q}$ denotes the beta(p, q) density and likelihood ratio is the ratio of the product of the $f(x_i|\theta_{z_i})$-terms for the new parameter set to that for the old. For the corresponding combine move, the acceptance probability is $\min(1, A^{-1})$, using the same expression for A but with some obvious difference in the substitutions.

The birth-and-death move [7] is simpler. We first make a random choice between birth and death, using the same probabilities b_M and d_M as above. For a birth, a weight and parameters for the proposed new component are drawn using

$$w_{j^*} \sim \mathcal{B}e(1, M), \qquad \mu_{j^*} \sim \mathcal{N}(\xi, \kappa^{-1}), \qquad \sigma_{j^*}^{-2} \sim \mathcal{G}(\alpha, \beta)$$

To provide space for the new component, the existing weights are rescaled, so that they all sum 1, using $w_j' = w_j(1 - w_{j^*})$. For a death, a random choice is made between any existing empty components, the chosen component is deleted and the remaining weights are rescaled to sum 1. No other changes are proposed to the variables: in particular, the allocations are unaltered.

We provide detailed balance holds for this move, assuming that we accept births and deaths according to expression (6), in which $(w_{j^*}, \mu_{j^*}, \sigma_{j^*}^2)$ play the role for $\boldsymbol{u}$. The use of the prior distributions in proposing values for μ_{j^*} and $\sigma_{j^*}^2$ leads to a simplification of the resulting ratio. The acceptance probabilities for birth and death are $\min(1, A)$ and $\min(1, A^{-1})$ respectively, where

$$\begin{aligned} A = & \frac{p(M+1)}{p(M)} \frac{1}{B(M\delta, \delta)} w_{j^*}^{\delta-1} (1 - w_{j^*})^{n+M\delta-M} (M+1) \\ & \times \frac{d_{M+1}}{(M_0+1) b_M} \frac{1}{g_{1,M}(w_{j^*})} (1 - w_{j^*})^M \end{aligned} \tag{11}$$

Here, M is the number of components and M_0 the number of empty components before the birth. In (11), the first line is the prior ratio, and the second line contains the proposal ratio and Jacobian; the likelihood ratio is 1.

2.3 Exponential mixtures[2]

We consider now the case of exponential mixtures. This may be relevant in CDR related fraud detection where we aim at modeling times between calls or call durations. We consider only the fixed mixture size case in this section.

Assume thus that we have a sample $\boldsymbol{x} = (x_1, \ldots, x_n)$ of data which are distributed according to a hyperexponential distribution, i.e., as a mixture of M exponentials. The generic parameter $\boldsymbol{\theta}$ is a vector containing the exponential parameters, $\boldsymbol{\theta} = \boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_M)$, and the general model (1) has then the expression

$$x_i \sim \sum_{j=1}^{M} w_j f_{\mathcal{E}xp}(x_i|\lambda_j),\ i = 1, \ldots, n,$$

where

$$f_{\mathcal{E}xp}(x_i|\lambda_j) = \lambda_j \exp(-\lambda_j x_i).$$

The motivation for modeling with such distribution resides in its flexibility and ability to cope with sample coefficients of variation bigger than one. More importantly, perhaps, we may use them to model hetereogeneity in the distribution of times, since we would be postulating M groups of customers, each group having different exponential time distributions. Our prior will be $\lambda_i \sim \mathcal{G}(a_i, b_i)$ and $\boldsymbol{w} \sim \mathcal{D}(\delta_1, \ldots, \delta_M)$, a Dirichlet distribution with parameters $(\delta_1, \ldots, \delta_M)$; $\delta_i > 0$. As with normal mixtures, the identifiability issue due to relabeling of parameters, is mitigated introducing the constraint $\lambda_1 < \cdots < \lambda_M$. This prior structure is largely chosen for its flexibility. Note that when $M = 1$, we simply have a Gamma prior distribution on the service rate which coincides with the usual conjugate choice. Under this model, the posterior distributions are essentially intractable. However, we can simulate a sample from the posterior distributions using a Markov Chain Monte Carlo (MCMC) method: after augmenting the data with indicators describing the element of the mixture each datum is coming from, we can set up a Gibbs sampler. We use an equivalent definition for the indicators, so that for each x_i, we define $\boldsymbol{z}_i = (z_{i1}, \ldots, z_{iM})$ with

$$\boldsymbol{z}_i|\boldsymbol{w}, \boldsymbol{\lambda} \sim \mathcal{M}(1; w_1, \ldots, w_M)$$

where $\mathcal{M}$ designates the multinomial distribution. Clearly, we have that $\sum_j z_{ij} = 1$, $z_{ij} \in \{0, 1\}$, and

$$x_i|\boldsymbol{z}_i, \boldsymbol{w}, \boldsymbol{\lambda} \sim \mathcal{E}xp\left(\prod_{j=1}^{M} \lambda_j^{z_{ij}}\right),$$

where $\mathcal{E}xp$ denotes the exponential distribution.

This allows us to specify the posterior conditionals given the other parameters, viz

$$\boldsymbol{z}_i|x_i, \boldsymbol{w}, \boldsymbol{\lambda} \sim \mathcal{M}\left(1; \frac{w_1\lambda_1 \exp(-\lambda_1 x_i)}{\sum_{j=1}^{M} w_j\lambda_j \exp(-\lambda_j x_i)}; \cdots; \frac{w_M\lambda_M \exp(-\lambda_M x_i)}{\sum_{j=1}^{M} w_j\lambda_j \exp(-\lambda_j x_i)}\right), i = 1, \ldots, n$$

[2]Based on Rios Insua et al. (1998), and French and Rios Insua (2000)

$$\lambda_j|\boldsymbol{x},\boldsymbol{z} \sim \mathcal{G}\left(a_j + \sum_{i=1}^{n} z_{ij}x_i, b_j + \sum_{i=1}^{n} z_{ij}\right), j = 1,\ldots,M$$

$$\boldsymbol{w}|\boldsymbol{x},\boldsymbol{z} \sim \mathcal{D}\left(\delta_1 + \sum_{i=1}^{n} z_{i1},\ldots,\delta_M + \sum_{i=1}^{n} z_{iM}\right)$$

Then, if $\boldsymbol{y}^t = (\boldsymbol{w}^t, \boldsymbol{z}^t, \boldsymbol{\lambda}^t)$ designates the state vector at the t-th iteration, the sampler goes through the following steps:

Algorithm 2.2. Gibbs sampler for fixed size exponential mixtures

i. Start with arbitrary values $(\boldsymbol{w}^0, \boldsymbol{\lambda}^0, \boldsymbol{z}^0)$, $t = 0$.

ii. Until convergence, iterate through

1. Generate $\boldsymbol{z}_i^{t+1} \sim \boldsymbol{z}_i|x_i, \boldsymbol{w}^t, \boldsymbol{\lambda}^t$, $i = 1,\ldots,n$.
2. Generate $\boldsymbol{w}^{t+1} \sim \boldsymbol{w}|\boldsymbol{x}, \boldsymbol{z}^{t+1}$.
3. Generate $\lambda_j^{t+1} \sim \lambda_j|\boldsymbol{x}, \boldsymbol{z}^{t+1}$, $j = 1,\ldots,M$.
4. Set $t = t + 1$.

2.4 Gamma mixtures[3]

We consider now modeling with gamma mixtures with both fixed and random mixture sizes. This may be appropriate in fraud detection when we need to deal with features describing a transaction which are continuous and positive and modes are not at zero.

Now the data $\boldsymbol{x} = (x_1,\ldots,x_n)$ come from a mixture of M gamma distributions. The generic parameter $\boldsymbol{\theta}$ is a vector of pairs (ν_j, γ_j), $j = 1,\ldots,M$, and the general model (1) takes the form

$$x_i \sim \sum_{j=1}^{M} w_j f_{\mathcal{G}}(x_i|\nu_j,\gamma_j),\ i = 1,\ldots,n,$$

where

$$f_{\mathcal{G}}(x_i|\nu_j,\gamma_j) = \frac{(\nu_j/\gamma_j)^{\nu_j}}{\Gamma(\nu_j)} x_i^{\nu_j - 1} \exp\left(-\frac{\nu_j}{\gamma_j}x_i\right).$$

Our parameterisation of each gamma density is in terms of its mean γ_j and shape parameter ν_j.

For the remaining model parameters, we shall assume proper but relatively diffuse distributions of the following form:

$$\begin{aligned}
\boldsymbol{w}|M &\sim \mathcal{D}(\delta_1,\ldots,\delta_M) \qquad \text{a Dirichlet distribution}\\
\nu_j|M &\sim \mathcal{E}xp(\rho) \qquad \text{an exponential distribution, for } j = 1,\ldots,M\\
\gamma_j|M &\sim \mathcal{GI}(\zeta,\varsigma) \qquad \text{an inverted gamma distribution, for } j = 1,\ldots,M
\end{aligned}$$

[3]Based on Wiper et al. (2001)

with the usual restriction $\gamma_1 < \cdots < \gamma_M$ to avoid unidentifiability. Typically, we might set $\delta_j = 1,\ j = 1, \ldots, M$ to give a uniform prior over the weights, $\rho = .01$, $\zeta = 1$ and $\varsigma = 1$. Note that the usual improper reference priors are inappropriate here, leading to improper posteriors: see Roeder and Wasserman (1997). Note that given $\zeta \leq 1$, the prior moments of $\boldsymbol{\gamma}$ do not exist. Thus, it is also easy to prove that, for $M > 1$, the posterior moments do not exist: if we are interested in estimating the mixture parameters rather than just modeling the distribution of the data, it may well be more appropriate to use a larger value of ζ.

The restriction on the prior of $\boldsymbol{\gamma}$ is used to ensure identifiability of the posterior distribution although it has the disadvantage of, theoretically, not allowing us to model mixtures of distributions with equal means. In practice, this has not proved to be a problem, see Wiper et al. (2001). We could use instead a lexicographic ordering procedure, supposing that $\gamma_j \leq \gamma_{j+1}$ and that if $\gamma_j = \gamma_{j+1}$ then $\nu_j < \nu_{j+1}$, but inference under this scheme would be somewhat harder to implement and for practical purposes would seem unlikely to improve the estimates very much. We note that the identifiability restriction is unnecessary if our interest is solely in modeling the distribution of the observable X.

As before, we want to undertake inference for the parameters of the mixture model. Suppose initially that the number of elements of the mixture M is known. We again introduce indicator variables as in (2) and use a hybrid sampler. If we use the same definition of the labels $z_i,\ i = 1, \ldots, n$ than in the normal mixtures, the conditional posterior distributions of the relevant model parameters given M are easily shown to be

$$\Pr\{z_i = j|\boldsymbol{x}, M, \boldsymbol{w}, \boldsymbol{\gamma}, \boldsymbol{\nu}\} \propto w_j \frac{(\nu_j/\gamma_j)^{\nu_j}}{\Gamma(\nu_j)} x_i^{\nu_j - 1} \exp\left(-\frac{\nu_j}{\gamma_j} x_i\right),$$

$$\boldsymbol{w}|\boldsymbol{x}, M, \boldsymbol{z}, \boldsymbol{\gamma}, \boldsymbol{\nu} \sim \mathcal{D}(\boldsymbol{\delta} + \mathbf{n}),$$

where $\boldsymbol{n} = (n_1, \ldots, n_M)$, with $n_j = \#\{i : z_i = j\},\ j = 1, \ldots, M$, and $\sum_{j=1}^{M} n_j = n$,

$$\gamma_j|\boldsymbol{x}, M, \boldsymbol{z}, \boldsymbol{w}, \boldsymbol{\nu} \sim \mathcal{GI}(\zeta + n_j\nu_j, \varsigma + S_j\nu_j),$$

where $S_j = \sum_{i:z_i=j} x_i$,

$$f(\nu_j|\boldsymbol{x}, M, \boldsymbol{z}, \boldsymbol{w}, \boldsymbol{\gamma}) \propto \frac{\nu_j^{n_j\nu_j}}{\Gamma(\nu_j)^{\nu_j}} \exp\left(-\nu_j\left(\theta + \tfrac{S_j}{\gamma_j} + n_j \log\gamma_j - \log P_j\right)\right), \quad (12)$$

where $P_j = \prod_{i:z_i=j} x_i$.

Thus, conditional on M, we can define a hybrid sampling algorithm to sample the full posterior distribution as follows:

The only complicated step is that of sampling the distribution of ν_j, for $j = 1, \ldots, M$. It is easy to show, however, that for large ν_j, the density given by Equation (12) is similar to a gamma density, and thus we use a Metropolis step with a gamma candidate distribution to generate candidate values. For

Algorithm 2.3. Hybrid sampler for gamma mixtures, fixed size

i. Start with arbitrary values $\boldsymbol{w}^0, \boldsymbol{\gamma}^0, \boldsymbol{\nu}^0,\ t = 0$.

ii. Until convergence, iterate through

1. Generate $\boldsymbol{z}^{t+1} \sim \boldsymbol{z}|\boldsymbol{x}, M, \boldsymbol{w}^t, \boldsymbol{\gamma}^t, \boldsymbol{\nu}^t$.
2. Generate $\boldsymbol{w}^{t+1} \sim \boldsymbol{w}|\boldsymbol{x}, M, \boldsymbol{z}^{t+1}, \boldsymbol{\gamma}^t, \boldsymbol{\nu}^t$.
3. Generate $\gamma_j^{t+1} \sim \gamma_j|\boldsymbol{x}, M, \boldsymbol{z}^{t+1}, \boldsymbol{w}^{t+1}, \boldsymbol{\nu}^t$ for $j = 1, \ldots, M$.
4. Generate $\nu_j^{t+1} \sim \nu_j|\boldsymbol{x}, M, \boldsymbol{z}^{t+1}, \boldsymbol{w}^{t+1}, \boldsymbol{\gamma}^{t+1}$ for $j = 1, \ldots, M$.
5. Order $\boldsymbol{\gamma}^{t+1}$ and sort $\boldsymbol{w}^{t+1}$ and $\boldsymbol{\nu}^{t+1}$ accordingly.
6. Set $t = t + 1$.

each $j = 1, \ldots, M$ we generate a candidate $\tilde{\nu}_j \sim \mathcal{G}(r, r/\nu_j^t)$ and accept this with probability

$$\min\left\{1, \frac{f(\tilde{\nu}_j|\ldots)p(\tilde{\nu}_j, \nu_j^t)}{f(\nu_j^t|\ldots)p(\nu_j^t, \tilde{\nu}_j)}\right\}$$

where $p(\tilde{\nu}, \nu)$ is the gamma density used to generate $\tilde{\nu}$ and $f(\tilde{\nu}_j|\ldots)$ is as in Equation (12). If the candidate is rejected, we retain the current value. The parameter r of the gamma proposal distribution may be adjusted to give a satisfactory acceptance rate.

Suppose now that M is unknown and that we define a prior density $p(M)$ with support $1, \ldots, M_{\max}$. Typically, we choose $M_{\max} < n$ although, in theory, the support for M could be infinite when we would have full semi-parametric modeling. Obviously, if given data, the posterior probability that M is equal to $M_{\max}$ is high, this would suggest that $M_{\max}$ should be increased. In practice, we should consider discrete uniform, truncated geometric or truncated shifted Poisson distributions. Then we can extend **Algorithm 2.3** by introducing a reversible jump to allow us to move through the posterior distribution of M. This is basically a Metropolis step which allows for a change in the dimension of the parameter space.

We modify then **Algorithm 2.3** by replacing M with M^t throughout and rewriting step [6] as

Algorithm 2.3 (extended) . RJ sampler for variable gamma mixtures

6a. Generate M^{t+1} and modify the other model parameters appropriately.

6b. Set $t = t + 1$.

The reversible jump algorithm proceeds as follows. Firstly, given a current

value of M we generate a candidate $\tilde{M}$:

$$\tilde{M} = \begin{cases} M+1 & \text{if } M = 1 \\ M-1 & \text{if } M = M_{\max} \\ M-1 \text{ with } p = 0.5 \text{ or } M+1 \text{ with } p = 0.5 & \text{if } 1 < M < M_{\max} \end{cases}$$

Given $\tilde{M}$, we now modify the other model parameters appropriately.

If $\tilde{M} = M - 1$, we must reduce the mixture size and thus we choose two neighboring terms in the mixture $(j1, j2 = j1 + 1)$ to combine, at random, with probability $p = 1/(M-1)$. Supposing that $\tilde{w}$, $\tilde{\gamma}$ and $\tilde{\nu}$ are the parameters of the combined term, we define:

1. $\tilde{w} = w_{j1} + w_{j2}$
2. $\tilde{w}\tilde{\gamma} = w_{j1}\gamma_{j1} + w_{j2}\gamma_{j2}$. and
3. $\tilde{w}\tilde{\gamma}^2\left(1 + \frac{1}{\tilde{\nu}}\right) = \sum_{i=1}^{2} w_{ji}\gamma_{ji}^2\left(1 + \frac{1}{\nu_{ji}}\right)$

Using these transformations, we preserve the restriction that the weights sum to 1 and the conditional moments

$$E[X^i|M, \ldots] = E[X^i|\tilde{M}, \ldots]$$

for $i = 0, 1, 2$.

The algorithm for increasing M to $M + 1$ is partially determined by the previous step. We now elect an element j of the mixture to split into two, with probability $1/M$. We then split the parameters as follows.

1. Generate $u_1 \sim U(0, 1)$ and define

$$\tilde{w}_{j1} = u_1 w_j, \quad \tilde{w}_{j2} = (1 - u_1)w_j$$

2. Generate $u_2 \sim U(0, 1)$ and define

$$\begin{aligned} \tilde{\gamma}_{j1} &= \gamma_{j-1} + u_2(\gamma_j - \gamma_{j-1}) \\ \tilde{\gamma}_{j2} &= \frac{(1 - u_1u_2)\gamma_j - u_1(1 - u_2)\gamma_{j-1}}{1 - u_1} \end{aligned}$$

   where the formula when $j = 1$ may be derived by setting $\gamma_0 = 0$ in the above.

3. Generate $u_3 \sim \mathcal{G}(s, s)$ and $\tilde{\nu}_{j1} = u_3\nu_j$ and $\tilde{\nu}_{j2}$ in consequence to satisfy Step 3 of the combination algorithm above. The parameter s can be adjusted by trial and error in order to achieve a satisfactory acceptance rate.

Given the generated parameter values $\boldsymbol{\theta}$, we now choose whether to accept or reject the move. We first note that in Steps 2 and 3 of the splitting algorithm, we may produce impossible values of $\tilde{\gamma}_{j2}$ and $\tilde{\nu}_{j2}$. If this should happen, we

would reject the move from M to $\tilde{M}$ immediately. Otherwise, we would accept it with probability

$$\min\left\{1, \frac{f(\tilde{\boldsymbol{\theta}})p_{\bar{m}}(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta})}{f(\boldsymbol{\theta})p_m(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}})}\right\}$$

where $p_m(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}})$ is the density of the move from $\boldsymbol{\theta}$ to $\tilde{\boldsymbol{\theta}}$ and $f(\boldsymbol{\theta})$ is the distribution of $\boldsymbol{\theta}$ given by

$$f(\boldsymbol{\theta}) \propto M! \left[\prod_{i=1}^{n}\left(\sum_{j=1}^{M} w_j \mathcal{G}(x_i|\nu_j, \nu_j/\gamma_j)\right)\right] f(M, \boldsymbol{w}, \boldsymbol{\gamma}, \boldsymbol{\nu}) \tag{13}$$

The densities of the moves are

$$p_{\text{combine}} = \frac{1}{2}\frac{1}{M-1}$$

and

$$p_{\text{split}} = \frac{1}{2}\frac{1}{M} f(u_3) \times \text{inv}$$

where $f(u_3)$ is the gamma density used to generate u_3 and inv is the Jacobian of the inverse transformation.

This algorithm is similar to that given in 2.2 for normal mixtures. The main difference is that we do not redefine the indicator variables z_i, within the reversible jump algorithm. Note that this is done at the very next step of the Gibbs sampling algorithm anyway so is not a problem. The cost of this is that the density given in Equation (13) is more complicated, including the full mixture likelihood.

2.5 Hierarchical mixture models[4]

We describe now a multilevel mixture model, in the normal case. This could be very relevant in fraud detection when we observe clusters of clusters of transactions. We limit the discussion to two levels, with extensions to more levels in the hierarchy being relatively straightforward. We assume we have multivariate data x_i.

At the level of the observed data, we induce clustering by assuming a mixture of (bivariate) normal models with an unknown number M of terms

$$x_i \sim \sum_{j=1}^{M} w_j \, \mathcal{N}(x_i|\mu_j, \Sigma_j), i = 1, \ldots, n,$$

where μ_j is the vector of means, and Σ_j is the covariance matrix for cluster j. The implied clustering is easiest seen in the equivalent formulation using latent indicator variables z_i with

$$\begin{aligned} \Pr\{z_i = j\} &= w_j, \\ p(x_i|z_i = j) &\sim \mathcal{N}(\mu_j, \Sigma_j), \end{aligned} \tag{14}$$

[4]Based on Cano et al. (2011)

The latent indicators z_i define clusters $\Gamma_j = \{i : \ z_i = j\}$, with size $n_j = \#\{i : z_i = j\} = |\Gamma_j|$.

At a second level, we assume that cluster locations μ_j and covariance matrices Σ_j are themselves clustered by a similar process

$$\theta_j = (\mu_j, \Sigma_j) \sim \sum_{k=1}^{L} v_k \, \mathcal{W}(\Sigma_j^{-1}|\nu, (\nu \, S_k)^{-1}) \, \mathcal{N}(\mu_j|\beta_k, \rho \, \Sigma_j), j = 1, \ldots, M,$$

where $\sum_{k=1}^{L} v_k = 1$, $v_k > 0$ are the weights, and $\mathcal{W}$ is an inverse Wishart distribution. Using latent indicators $s_j = 1, \ldots, L$ we can rewrite the mixture

$$\begin{aligned} \Pr\{s_j = k\} &= v_k, \\ p(\mu_j, \Sigma_j|s_j = k) &\sim \mathcal{W}\big(\Sigma_j^{-1}|\nu, (\nu \, S_k)^{-1}\big) \, \mathcal{N}\big(\mu_j|\beta_k, \rho \, \Sigma_j\big), \end{aligned} \tag{15}$$

As in the top level mixture, the latent indicators s_j define super-clusters $\Delta_k = \{j : \ s_j = k\}$, of size $m_k = \#\{j : s_j = k\} = |\Delta_k|$.

The model is completed with priors on the parameters, including the size M and L of the mixtures, the weights $\boldsymbol{w} = (w_1, \ldots, w_M)$ and $\boldsymbol{v} = (v_1, \ldots, v_L)$ and the parameters of the second level mixture terms β_k and S_k. For the mixture sizes, we use a prior distribution with positive probability on $M = L = 1$, implying prior positive probability for a simpler embedded model with one or no mixture levels.

Specifically, we assume geometric priors $p(M) \propto \alpha_1^M$, $p(L) \propto \alpha_2^L$, favoring parsimony a priori, assuming $L < M$. In case there is no uncertainty on L and/or M, we may assume degenerate priors for L and/or M.

We use Dirichlet priors for the weights, $p(\boldsymbol{w}|M) = \mathcal{D}(\delta_1, \ldots, \delta_M)$, $p(\boldsymbol{v}|L) = \mathcal{D}(\varphi_1, \ldots, \varphi_L)$; and conjugate priors $p(\beta_k) = \mathcal{N}(0, B^{-1})$, $p(S_k) = \mathcal{W}(s_H, S_H)$.

The hierarchical clustering structure implied by (14) and (15) is elucidated by considering the equivalent description as a nested mixture of mixtures. Let $t_i = s_{z_i}$ denote the super-cluster associated with observation i, and re-label the first-level clusters $j = 1, \ldots, M$ as $j_1 = 1, \ldots, m_1$, $j_2 = 1, \ldots, m_2$, $\ldots, j_L = 1, \ldots, m_L$, and the cluster parameters as $\mu_{11}, \ldots, \mu_{1,m_1}, \ldots, \mu_{L,m_L}$, etc. We can now rewrite the model as

$$\begin{aligned} \Pr\{t_i = k\} &= v_k, \ i = 1, \ldots, n \\ \Pr\{z_i = j|t_i = k\} &= w_{kj}, \ i = 1, \ldots, n \\ p(m_1, \ldots, m_L) &\propto \mathcal{M}(1; \varphi_1, \ldots, \varphi_L), \\ p(x_i|t_i = k, z_i = j) &\propto \mathcal{N}(x_i|\mu_{kj}, \Sigma_{kj}), \end{aligned} \tag{16}$$

and conjugate prior $p(\mu_{kj}, \Sigma_{kj}) = \mathcal{W}[\Sigma_{jk}^{-1}|\nu, (\nu \, S_k)^{-1}] \ \mathcal{N}(\mu_{jk}|\beta_k, \rho \, \Sigma_{jk})$. The indicators t_i define the super-clusters at the coarse level; and the z_i denote the j-th cluster within this super-cluster.

Assuming M, L fixed for now, the posterior sampling scheme would be

Specifically, the distributions involved are:

1. The conditional posterior for z_i is a multinomial (of size 1) satisfying $Pr(z_i = j) \propto w_j \times \mathcal{N}(x_i|\mu_j, \Sigma_j), \ i = 1, \ldots, n, \ j = 1, \ldots, M.$

Algorithm 2.4. Gibbs sampler for hierarchical mixtures, fixed sizes

i. Start with arbitrary values $\boldsymbol{w}^0, \boldsymbol{\mu}^0, \boldsymbol{\Sigma}^0, \boldsymbol{s}^0, \boldsymbol{v}^0, \boldsymbol{\beta}^0, \boldsymbol{S}^0, \boldsymbol{z}^0,\ t = 0$.

ii. Until convergence, iterate through

1. Generate $z_i^{t+1} \sim z_i|\boldsymbol{x}, M, \boldsymbol{\mu}^t, \Sigma_j^t,\ i = 1, \ldots, n$.
2. Generate $\boldsymbol{w}^{t+1} \sim \boldsymbol{w}|\boldsymbol{x}, \boldsymbol{z}^{t+1}, M$.
3. Generate $\mu_j^{t+1} \sim \mu_j|\boldsymbol{x}, \boldsymbol{z}^{t+1}, \Sigma_j^t, s_j^t = k, \beta_k,\ j = 1, \ldots, M$.
4. Generate $\Sigma_j^{t+1} \sim \Sigma_j|\boldsymbol{x}, \boldsymbol{z}^{t+1}, \mu_j^{t+1}, s_j^t = k, \beta_k,\ j = 1, \ldots, M$.
5. Generate $s_j^{t+1} \sim s_j|\boldsymbol{x}, L, \beta_M, S_M,\ j = 1, \ldots, M$.
6. Generate $\boldsymbol{v}^{t+1} \sim \boldsymbol{v}|\boldsymbol{x}, s^{t+1}, L$.
7. Generate $\beta_k^{t+1} \sim \beta_M|\boldsymbol{x}, s^{t+1}, \boldsymbol{\mu}^{t+1}, \Sigma_j^{t+1},\ k = 1, \ldots, L$.
8. Generate $S_k^{t+1} \sim S_M|\boldsymbol{x}, s^{t+1}, \Sigma_j^{t+1},\ k = 1, \ldots, L$.
9. Set t=t+1.

2. The conditional posterior for w is a Dirichlet with parameters $\delta_j + n_j,\ j = 1, \ldots, M$.

3. The conditional posterior μ_j is normal with mean vector $\left(n_j + \frac{1}{\rho}\right)^{-1} \left(n_j \bar{x}_j + \frac{1}{\rho}\beta_k\right)$ and covariance matrix $S = \Sigma_j \left(n_j + \frac{1}{\rho}\right)^{-1},\ j = 1, \ldots, M$

4. The conditional posterior for Σ_j^{-1} is $\mathcal{W}\left(n_j + \nu + 1; W\right)$ with

$$W = \nu S_k + \rho^{-1}\left(\mu_j - \beta_k\right)\left(\mu_j - \beta_k\right)' + V_j + n_j\left(\bar{x}_j - \mu_j\right)\left(\bar{x}_j - \mu_j\right)'$$

5. The conditional posterior of s_j, is a multinomial (of size 1) which satisfies $Pr(s_j = k) \propto v_k \times \mathcal{N}(\mu_j|\beta_k, \rho\Sigma_j)\mathcal{W}\left[\Sigma_j^{-1}|\nu, (\nu S_k)^{-1}\right],\ j = 1, \ldots, M,\ k = 1, \ldots, L$.

6. The conditional posterior of v is a Dirichlet with parameters $(\varphi_k + m_k),\ k = 1, \ldots, L$.

7. The conditional posterior distribution of $\beta_k,\ k = 1, \ldots, L$ is normal with mean vector

$$m = \left(B + \frac{1}{\rho}\sum_{j \in \Delta_k} \Sigma_j^{-1}\right)^{-1} \frac{1}{\rho}\sum_{j \in \Delta_k} \mu_j' \Sigma_j^{-1}.$$

and covariance matrix

$$S = \left(B + \frac{1}{\rho}\sum_{j \in \Delta_k} \Sigma_j^{-1}\right)^{-1}$$

8. The conditional posterior of S_k is $\mathcal{W}\left(s^H + \nu \cdot m_k; W\right)$ with

$$W = \left(\sum_{j \in \Delta_k} \nu \Sigma_j^{-1} + \left(S^H\right)^{-1} \right)^{-1}, \; k = 1, \ldots, L$$

Keeping random the size of the mixture, as in our case with L and M, considerably complicates posterior simulation. Rather than using a reversible jump (RJ) algorithm for Markov chain Monte Carlo simulation in such models, we shall appeal to Stephens (2000) method based on a birth death process which we shall adopt here to our hierarchical structure, due to its simpler analysis.

To wit, if $(M, \boldsymbol{w}, \boldsymbol{\phi}(= \boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{z})$ and $(L, \boldsymbol{v}, \boldsymbol{\psi}(= \boldsymbol{\beta}, \boldsymbol{S}), \boldsymbol{s})$ designate first and second level parameters, respectively, and $\boldsymbol{\theta} = (M, \boldsymbol{\phi}, L, \boldsymbol{\psi})$, we have the following extension to **Algorithm 2.4** to simulate $\boldsymbol{\theta}^{t+1}$, given $\boldsymbol{\theta}^t$:

Algorithm 2.4 (extended).
MCMC sampler for hierarchical mixtures, variable size

i. Start with arbitrary values $\boldsymbol{w}^0, \boldsymbol{\gamma}^0, \boldsymbol{\nu}^0, M^0$ $t = 0$.

ii. Until convergence, iterate through

0. Generate $(M^{t'}, \boldsymbol{w}^{t'}, \boldsymbol{\phi}^{t'})$ running a Stephens' like birth death process until time t_0, starting at $(M^t, \boldsymbol{w}^t, \boldsymbol{\phi}^t)$.

Set $(M^{t+1}, \boldsymbol{w}^{t+1}, \boldsymbol{\phi}^{t+1}) = (M^{t'}, \boldsymbol{w}^{t'}, \boldsymbol{\phi}^{t'})$.

1,2,3,4. Generate $\boldsymbol{z}, \boldsymbol{w}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ as before.

0'. Generate $(L^{t'}, \boldsymbol{v}^{t'}, \boldsymbol{\psi}^{t'})$ running a Stephens' like birth death process until time t_0, starting at $(L^t, \boldsymbol{v}^t, \boldsymbol{\psi}^t)$.

Set $(L^{t+1}, \boldsymbol{v}^{t+1}, \boldsymbol{\psi}^{t+1}) = (L^{t'}, \boldsymbol{v}^{t'}, \boldsymbol{\psi}^{t'})$.

5,6,7,8. Generate $\boldsymbol{s}, \boldsymbol{v}, \boldsymbol{\beta}, \boldsymbol{S}$ as before.

9. Set t=t+1.

For the first level changes, we use the first level likelihood

$$\mathcal{L}(M, \boldsymbol{w}, \boldsymbol{\phi}) = \pi_{i=1}^n \left[w_1 \mathcal{N}(x_i; \boldsymbol{\phi}) + \cdots + w_M \mathcal{N}(x_i; \phi) \right],$$

whereas for second level changes, we use the second level likelihood

$$\mathcal{L}(L, \boldsymbol{v}, \boldsymbol{\psi}) = \pi_{i=1}^n \left[v_1 \mathcal{N}(.,.) \mathcal{W}(.,.) + \cdots + v_L \mathcal{N}(\cdot,\cdot) \mathcal{W}(\cdot,\cdot) \right]$$

3 Using mixture models for fraud detection

We describe now how we may use the previous models for fraud detection.

3.1 Detecting the number of clusters

Recall that in the cluster analysis document, we mentioned that a key issue is determining the number of clusters. This may be addressed coherently within the previous framework as follows.

Once the reversible jump MCMC has been performed, we can obtain the most likely value of M. Specifically, the algorithm has the following steps:

Algorithm 3.1. Determining the number of clusters

i. From the sampler, obtain $\{\boldsymbol{y}^r = (M^r, \boldsymbol{z}^r, \boldsymbol{w}^r, \boldsymbol{\theta}^r)\}_{r=1}^N$.

ii. Approximate $p_j = \Pr\{j \ \ clusters|data\} \approx \dfrac{\#M^r = j}{N}$, for $j = 1, \ldots, M$

iii. Choose maxprob cluster $M^* = \{j : \max \ p_j\}$

Then, we could run an M^*-means algorithm and proceed as described in the previous document.

3.2 Classification

We describe now how we may use the above models for classification purposes. Here we would aim at classifying a transaction as fraudulent or not. More generally, we could consider several classes of fraudster behavior or normal behaviour. In this section, we shall assume that the number of classes is fixed, with some of them corresponding to fraud and others to normal behaviour. We assume that we may observe a training sample of some t perfectly classified cases, and a further u unclassified cases. Thus we assume we observe data $\boldsymbol{x}_{(T)} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t)$ together with classification indicators $\boldsymbol{z}_{(T)} = (z_1, \ldots, z_t)$, and then $\boldsymbol{x}_{(U)} = (\boldsymbol{x}_{t+1}, \ldots, \boldsymbol{x}_{t+u})$. We will process these two data sets sequentially, $(\boldsymbol{x}_{(T)}, \boldsymbol{z}_{(T)})$ followed by $\boldsymbol{x}_{(U)}$. We will then proceed to infer the model parameters $\boldsymbol{\theta}$ and the classification quantities $\boldsymbol{z}_{(U)}(z_j; j = t+1, \ldots, t+u)$ and also to predictive classification of future cases.

Consider first processing the training sample $(\boldsymbol{x}_{(T)}, \boldsymbol{z}_{(T)})$. The prior is conjugate and the analysis is standard, since the data are perfectly classified into normal components, the quantities $\boldsymbol{z}_{(T)} = (z_1, \ldots, z_t)$ being known. The structure of the joint posterior for $\boldsymbol{\theta}$ given $(\boldsymbol{x}_{(T)}, \boldsymbol{z}_{(T)})$ is just that of the prior, with the defining parameters appropriately updated, as e.g. described in French and Ríos Insua (2000).

Consider now the unclassified sample $\boldsymbol{x}_{(U)}$. $(\boldsymbol{x}_{(U)}, \boldsymbol{z}_{(U)})$ is conditionally independent of $(\boldsymbol{x}_{(T)}, \boldsymbol{z}_{(T)})$ given the parameters $\boldsymbol{\theta}$, and this involves simply replacing the prior for $\boldsymbol{\theta}$ throughout by the similarly structured distribution $p(\boldsymbol{\theta}|\boldsymbol{x}_{(T)}, \boldsymbol{z}_{(T)})$, that summarizes the revised state of information about the parameters based on the training sample.

Let $\boldsymbol{D} = (\boldsymbol{x}_{(T)}, \boldsymbol{z}_{(T)}, \boldsymbol{x}_{(U)})$ be the known data information. Then, Monte

Carlo approximations to $p(\boldsymbol{\theta}|\boldsymbol{D})$ and $p(\boldsymbol{z}_{(U)}|\boldsymbol{D})$ are given by the mixtures

$$\begin{aligned} p(\boldsymbol{\theta}|\boldsymbol{D}) &\simeq N^{-1}\sum_{r=1}^{N} p(\boldsymbol{\theta}|\boldsymbol{D}, \boldsymbol{z}_{(U)_r}) \\ p(\boldsymbol{z}_{(U)}|\boldsymbol{D}) &\simeq N^{-1}\sum_{r=1}^{N} p(\boldsymbol{z}_{(U)}|\boldsymbol{D}, \boldsymbol{\theta}_r) \end{aligned} \tag{17}$$

where $\{\boldsymbol{z}_{(U)_r}, \boldsymbol{\theta}_r\}_{r=1}^{N}$ are samples from the appropriate posteriors.

(i) The first equation in (17) has a margin for $\{\boldsymbol{\mu}_i\}_{i=1}^{M}$ that is a mixture of conditional T-posteriors, easily evaluated and summarized. Similarly, inference about $\boldsymbol{\Sigma}_i$ will be based on a mixture of inverse Wisharts.

(ii) The posterior probabilities $\Pr\{z_j = i|\boldsymbol{D}\}, i = t+1, \ldots, t+u$ are of particular interest in classification of fraudulent transactions. The Monte Carlo estimates of such quantities is given by the second equation in (17). We would just compute the maximum probability class for an unlabeled observation and if corresponding to a fraudulent class issue a warning. Note that this would correspond to a kind of forensic determination of fraud.

(iii) Regarding the prediction of further observations, suppose that an observation $\boldsymbol{x}_f$ is known to come from component i of the mixture; thus, if z_f is the classification indicator for $\boldsymbol{x}_f$, we require the density function $p(\boldsymbol{x}_f|\boldsymbol{D}, z_f = i)$. Now (17) applies to give a mixture of T-distributions.

(iv) If z_f is unknown, the density $p(\boldsymbol{x}_f|\boldsymbol{D}) \simeq N^{-1}\sum p(\boldsymbol{x}_f|\boldsymbol{D}, \boldsymbol{z}_{(U)_r})$ forms the basis for prediction of z_f. The mixture components are given by

$$p(\boldsymbol{x}_f|\boldsymbol{D}, \boldsymbol{z}_{(U)}) = \sum_{j=1}^{M} \Pr\left\{z_f = j|\boldsymbol{D}, \boldsymbol{z}_{(U)}\right\} p(\boldsymbol{x}_f|\boldsymbol{D}, \boldsymbol{z}_{(U)}, z_f = j)$$

The probability forming the first term of the summand here is evaluated as $\Pr\left\{z_f = j|\boldsymbol{D}, \boldsymbol{z}_{(U)}\right\} = E(\boldsymbol{\theta}_i|\boldsymbol{D}, \boldsymbol{z}_{(U)})$, and the second term is just the density in (iii).

(v) In attempting to classify $\boldsymbol{x}_f$ when z_f is unknown, we are interested in the posterior probabilities

$$\Pr\{z_f = j|\boldsymbol{D}, \boldsymbol{x}_f\} \propto \Pr\{z_f = j|\boldsymbol{D}\} \Pr\{\boldsymbol{x}_f|\boldsymbol{D}, z_f = j\} \tag{18}$$

The first term here is simply approximated, using (17), as $\Pr\{z_f = j|\boldsymbol{D}\} = E(\boldsymbol{\theta}_i|\boldsymbol{D}) \simeq N^{-1}\sum E(\boldsymbol{\theta}_i|\boldsymbol{D}, \boldsymbol{z}_{(U)_r}))$ the sum over $r = 1, \ldots, N$. The second term is evaluated as in (iii). Again, we would identify the class of maximum predictive probability and, if corresponding to a fraudulent one, issue a warning.

3.3 Outlier detection

Again, we shall consider in this discussion that the number of terms in the mixture is fixed and we are dealing with normal mixtures. Extensions to other cases is straightforward. Outlier detection is relevant in fraud detection if we assimilate fraudulent behavior with behavior very different to standard behavior, which we might assimilate with normal behavior at some clusters or standard fraud behavior at fraud clusters. What we now describe could be composed with the previous approach, as we describe.

Outlier detection will be based on the fact that clustering relies on weighted Mahalanobis distance, $\Pr\{r_i = j|x_i\} \propto q_j \, \mathcal{N}(x_i|\mu_j, \Sigma_j)$. Therefore, we shall declare an observation anomalous (and therefore possibly associated with a fraudulent transaction) when the corresponding indices are small enough, which could be implemented through the following empirical rules:

- We would say that an observation x_i is anomalous if it is "far enough" from all the centers $\mu_j^*,\ j = 1, \ldots, M$, i.e.,

  $$|x_i - \mu_j^*| \geq K\sigma_j^*,\ j = 1, \ldots, M$$

  where (μ_j^*, σ_j^*) are estimates of the centers and standard deviations for the j-th cluster, and K is typically chosen as 2, 3, 4,..., depending on the rigorousness of the filter we want to incorporate. A specific choice should take into account utility issues as explained in Section 4.

- Similarly, if the (posterior) probability of belonging to a certain group w_i^* multiplied by the corresponding normal density $f(x_i|\mu_j^*, \sigma_j^{*2})$ is "small enough" for all groups, i.e.,

  $$w_i^* f(x_i|\mu_j^*, \sigma_j^{*2}) \leq \epsilon, \text{ for all } j$$

  and for a sufficiently small positive constant ϵ, regulated by the user. Again, a specific choice of ϵ should take into account utility issues as explained in Section 4.

As for combinations with the previous approach, we could proceed with a rule as follows:

Algorithm 3.2. Rule for fraud warning issue

```
i. If maxprob cluster fraudulent, issue a warning.

ii. Else if it lays too far from the inferred mean behavior
 (which would be normal), issue a warning.

iii. Else, issue no warning.
```

3.4 Online fraud detection

Based on the previous approaches, we provide a general scheme which takes into account the dynamic behavior of the problem as data are available and, more specifically, as data concerning fraudulent behavior is available. We consider an architecture in which a *backoffice* server processes daily data and passes periodically the relevant models to *frontoffice* servers which provide online response to transactions declaring them as normal or suggesting caution because of possible fraudulent behavior.

Algorithm 3.3. Architecture for online fraud detection

i. BACKOFFICE (At beginning of period t)

1. Form $(\boldsymbol{x}^t_{(T)}, \boldsymbol{z}^t_{(T)}, \boldsymbol{x}^t_{(U)})$ from $(\boldsymbol{x}^{t-1}_{(T)}, \boldsymbol{z}^{t-1}_{(T)}, \boldsymbol{x}^{t-1}_{(U)})$ and new data available.
2. Process $(\boldsymbol{x}^t_{(T)}, \boldsymbol{z}^t_{(T)}, \boldsymbol{x}^t_{(U)})$.
3. Find M^*_t.
4. Send current model to front office servers.

ii. FRONTOFFICE (During period t)

1. For each new transaction,
   a. If maxprob cluster is fraudster, issue a warning
   b. Else if too far from mean, issue a warning.
2. Accumulate data.

4 Relevance of the utility function

Statistically based approaches to fraud detection, see Bolton and Hand (2002) tend to center over minimizing the number of false positives and negatives detected, without paying much attention to the evaluation of such errors or using a very simplistic cost function. From our point of view, forecasting such costs is important as it impacts on the utility function of the anti-fraud decision maker, and usually decision makers are not well prepared to deal with uncertainty measures associated with predictions. We illustrate its relevance.

4.1 A generic model for antifraud management[5]

Figure 1 shows an *influence diagram* (see Pearl (2005) or French and Rios Insua (2000)) displaying the simplest version of the antifraud management problem. The antifraud decision a (typically stopping the transaction to investigate it or not, although continuous decisions such as those based on a moving window,

[5]Based on Moreno and Rios Insua (2009)

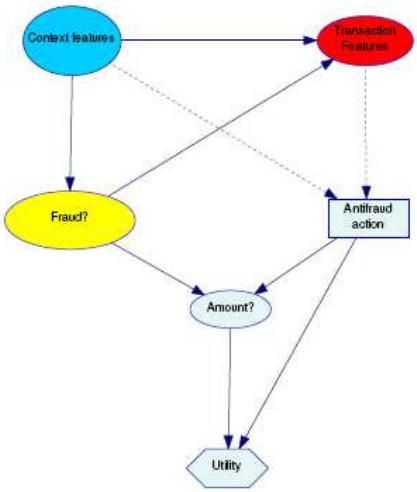

Figure 1: Basic antifraud influence diagram.

like the mean amount for the last 15 days, could be envisaged), is made with two types of data available: the transaction data t (referring to how the transaction is performed, how big is the transaction,...) and context data c (referring, e.g., to where the transaction originates, the sex of the sender,...), and before knowing whether there is actually a fraudulent transaction f. A key variable will be the amount defrauded m, which depends on the decision and on whether there is fraud or not. The utility function $u(m, a)$ depends on the amount defrauded and the decision made, to possibly include psychological factors such as regret or pride. As described in the diagram, the relevant distributions would be $p(c)$, $p(f|c)$, $p(t|f, c)$ and $p(m|f, a)$, reflecting the contextual dependencies.

We aim at determining the optimal decision a by computing the maximum expected utility action, for given c, t, see e.g. French and Ríos Insua (2000), which, in this case, would amount to solving the problem

$$\max_a \iint u(m, a)p(m|f, a)p(f|c, t)\ dm\ df, \tag{19}$$

and would provide the optimal decision given the data c, t available. Note that the initial formulation, as expressed in Figure 1 refers to the most natural one, in terms of obtaining the probabilities from experts or from a datawarehouse or from the type of models previously explained and we just perform some convenient computations to facilitate decision making.

Of course, we shall actually need to deal with more complex problems, as those in previous reports or Section 2. As an example, Figure 2 describes a model for assessing on *Bad Debt*, in which we essentially expand the context variables: the *Customer Type* node extended with *Credit Bureau Information*, and the *Fraud Behavior* node extended by variables extracted from data mining

process, so as to include all variables which appear to be relevant in this problem.

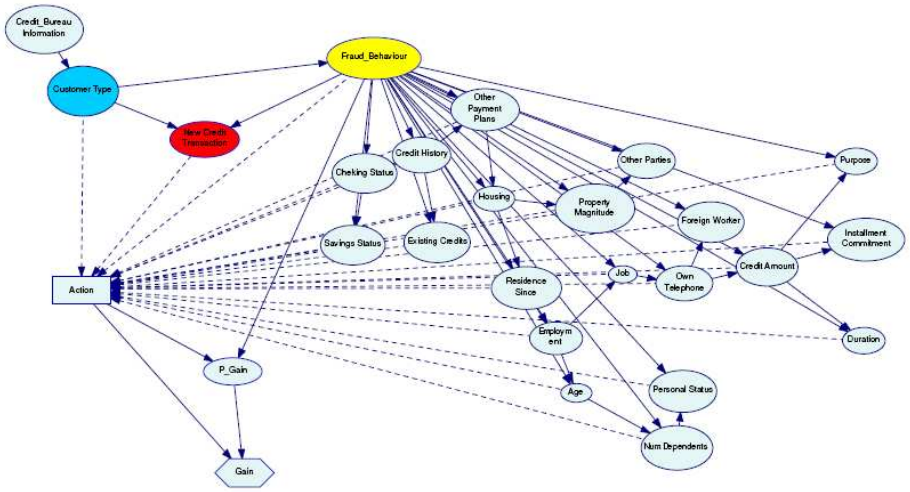

Figure 2: Influence diagram for the extended bad debt problem.

Specifically, the *Fraud Behavior* node expresses the probability of data corresponding to good or bad fraud behavior, which entails a more complex submodel which includes variables such as *Credit History*, *Checking Status*, *Savings Status*, *Existing Credits*, *Other Payment Plans*, *Housing, Property Magnitude*, *Job*, etc., as derived from using existing data mining technology to automate probability computations, based on the data provided.

We describe in detail a version of our basic example on credit debt to illustrate numerical subtleties and the relevance of the issues addressed. We consider a discrete problem to facilitate its implementation with standard software such as Genie, see `http://genie.sis.pitt.edu/`

Our basic assessments are as follows. Assume that, for the customer profile node, we have assessed

$$\Pr\{\text{Customer Profile: Clean}\} = 0.8$$

and

$$\Pr\{\text{Customer Profile: Alerting Fraud}\} = 0.2.$$

The *Credit Card Transaction* probabilities are shown in Table 1
whereas the *Fraud behavior* or *Card Fraud Type* node include the following probabilities, as shown in Table 2
The *Cost Chance* node probabilities are assessed as shown in Table 3, where we have considered three cost levels

In this example, we force the utility function to depend on both the cost and the decision made as expressed in Table 4

Fraud Behavior	Clean	Clean	Fraud Type	Fraud Type
Customer Profile	Clean	Fraud	Clean	Fraud
Bad	0.1	0.3	0.4	0.7
Good	0.9	0.7	0.6	0.3

Table 1: Credit Card Transaction node

Customer Profile	Clean	Fraud
Clean	0.8	0.2
FraudType	0.2	0.8

Table 2: Fraud Behavior node

Fraud Behavior	Clean	Clean	FraudType	FraudType
Action	Commit	Cancel	Commit	Cancel
None	0.8	0.9	0.1	1.0
Mid damage	0.1	0.075	0.3	0.0
Big damage	0.1	0.025	0.6	0.0

Table 3: Cost Chance node

This business context would essentially refer to a Service Level Agreement situation in which the costs of canceling if no fraud is present are always constant. Specifically, this utility function means that:

- If we accept that there is no fraud and this is true, there is no extra cost. However, if finally our prediction is wrong (false negative) and there is fraud, we are exposed to suffer a potential fraud defined by our previous company experience. We provide two reference points of those experiences: mean and biggest fraud history amount.
- If we choose to cancel the transaction, and finally there is fraud, we do not suffer any extra cost (we have completed satisfactorily our anti-fraud work), but if our prediction is wrong and there is actually no fraud, we incur extra costs due to the loss from company image (customer churning, company credibility), and low service quality (costs from wasted resources in additional activities with no added value) . We use two measures for quantifying our previous experiences with false positives: mean and biggest costs.

With such assessments, the optimal decisions are:

- *Cancel*, if *Customer Profile: Clean* and *Card Transaction: Bad* with -250 (vs. -316);

	None	Mid	Big
Accept	0	-5	-900
Deny	0	-5000	-5000

Table 4: Utility function for basic model

- *Accept*, if *Customer Profile: Clean* and *Card Transaction: Good* with -154 (vs. -428);
- *Cancel*, if *Customer Profile: Alert Fraud* and *Card Transaction: Bad* with -48 (vs. -497);
- *Cancel*, if *Customer Profile: Alert Fraud* and *Card Transaction: Good* with -184 (vs. -375).

We consider now a sensitivity analysis over the Credit Card model with respect to the utility function, by running the model with different utility inputs, but fixed probability inputs. We could use this as a *what if* analysis, the user varying some parameters or changing some assumptions. On one hand, we emphasize the need to provide an adequate economical evaluation of the problem, moving away by just minimizing performance measures related with false positives and negatives; on the other hand, we show that by introducing several utility functions, we provide a mechanism to segment the market.

Specifically, we could develop anti-fraud models for different customer segments. We just consider the case of reviewing a black list for existing card fraudsters, a situation in which performing anti-fraud actions generates constant costs, as expressed through the following utility function expressed in Table 5

	None	Mid	Big
Commit	0	-100	-10000
Cancel	-500	-500	-500

Table 5: Scenario 1

The optimal action is always *Cancel*.

5 Application to CDR's

We now sketch how the above approach may be used to deal with CDR's. Generally speaking, a CDR is associated with the performance of a user and is composed of data with the following structure (starting time of call, duration of call, type of call). Assume, for example, that type of call may be of three types (say, local, national, international).

In the simplest modeling scenario we would need to assess:

- The proportion of various types of calls p_1, p_2, p_3.
- The duration of i-th type calls, which we assume $\mathcal{E}(\mu_i)$, and independent.
- The distribution of time between calls[6], which we assume $\mathcal{E}(\mu_4)$, and independent.

Variants could include nonhomogeneity of the exponential rate over time, nondependence of the exponential distribution and others.

We would then assimilate the CDR to:

- $(p_1, p_2, p_3) \sim \mathcal{D}(a_1, a_2, a_3)$.
- $(\mu_1, \mu_2, \mu_3, \mu_4) \sim (\mathcal{G}(\alpha_1, \beta_1), \mathcal{G}(\alpha_2, \beta_2), \mathcal{G}(\alpha_3, \beta_3), \mathcal{G}(\alpha_4, \beta_4)$

and, therefore, associate with each CDR the vector $(a_1, a_2, a_3, \alpha_1, \beta_1, \alpha_2, \beta_2, \alpha_3, \beta_3, \alpha_4, \beta_4)$. As data is available we update them to $(a_1^*, a_2^*, a_3^*, \alpha_1^*, \beta_1^*, \alpha_2^*, \beta_2^*, \alpha_3^*, \beta_3^*, \alpha_4^*, \beta_4^*)$. With such approach we can apply the previously described approach based on mixtures.

As a final issue, note that there would be another relevant fraud problem which would correspond, e.g., to the case in which a mobile phone is stolen from a standard user. Detecting this is again an outlier detection problem: based on the current CDR model, check whether the new CDR data separates too much from the expected behaviour, much as described in Section 3.3.

6 Conclusions

We have described an improved framework for fraud detection. We have emphasized the following issues:

- Bayesian mixture modeling to capture heterogeneity in a population, with possibly some of the groups associated with fraudster.
- Coherent procedures to issue warnings in relation with a possible fraudulent transaction.
- The need of properly evaluating economically fraud detection.
- How CDR data may be dealt with in such framework.

Acknowledgments

Research supported by grants from RIESGOS-CM program S2009/ESP-1685.

[6]This refers to the time between finishing a call and starting the next one

References

R. J. Bolton and D. J. Hand. Statistical Fraud Detection: A Review. *Statistical Science*, 17(3):235–249, 2002.

S. French and D. Ríos Insua. *Statistical Decision Theory*. Arnold, 2000.

P. J. Green. Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination. *Biometrika*, 82(4):711–732, 1995.

J. Pearl. Influence Diagrams–Historical and Personal Perspectives. *Decision Analysis*, 2(4):232–234, 2005.

K. Roeder and L. Wasserman. Practical Bayesian Density Estimation Using Mixtures of Normals. *Journal-American Statistical Association*, 92(439):894–902, 1997.

M. Stephens. Bayesian Analysis of Mixture Models with an Unknown Number of Components-an Alternative to Reversible Jump Methods. *Annals of Statistics*, 28(1):40–74, 2000.

M. Wiper, D. Ríos Insua, and F. Ruggeri. Mixtures of Gamma Distributions with Applications. *Journal of Computational and Graphical Statistics*, 10(3): 440–454, 2001.