1 Introdução

Utilizando o tutorial Identifying Bayesian Mixture Models como referência do nosso estudo de caso, e os dados presentes no dataset de músicas do Spotfiy. Iremos investigar a dinâmica e os desafios de se modelar a distribuição da popularidade de uma amostra de músicas através de uma mistura de Gaussianas utilizando a biblioteca RStan.

2 Modelo teórico

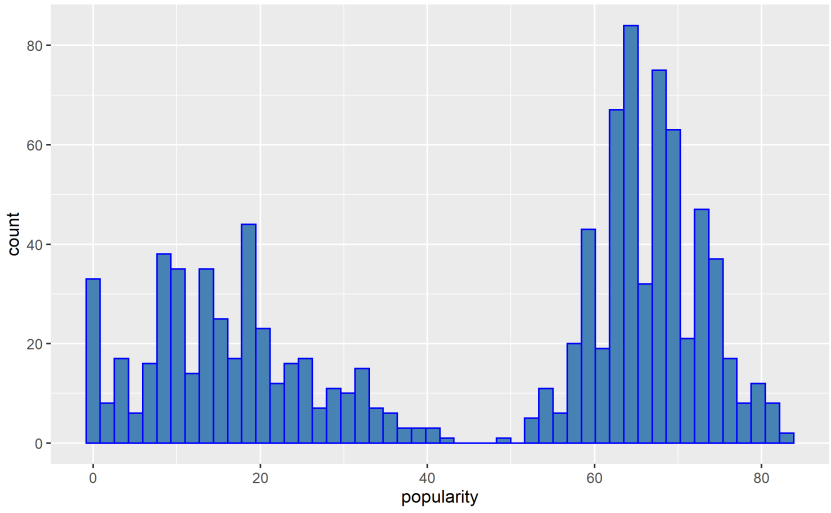
A seguir temos a verossimilhança como uma de mistura de K Gaussianas, onde cada peso θ_k pode ser interpretado como uma medida de importância de cada gaussiana, onde $\sum_{k=1}^{K} \theta_k = 1$.

$$\pi(y_1, ..., y_N | \vec{\mu}, \vec{\sigma}, \vec{\theta}) = \sum_{n=1}^{N} \sum_{k=1}^{K} \theta_k \cdot \mathcal{N}(y_n | \mu_k, \sigma_k)$$

Se fizéssemos uma inferência usando esta versão, precisaríamos simular cada componente dos pesos θ_k e dos parâmetros μ_n e σ_n. Tornando-se um problema computacionalmente custoso.

Além disso, se considerar que, neste caso, não há discriminação entre qual gaussiana deve fitar qual cluster de dados, essa ambiguidade traria um total de $K!$ possíveis jeitos de discriminar cada gaussiana a cada cluster. Portanto, fazendo provável o colapso de qualquer programa em tentar seguir por esta abordagem custosa.

Como nosso exemplo de estudo, a seguir temos o histograma de uma amostra de 1000 músicas segmentadas quanto a sua porcentagem de popularidade na plataforma do Spotfiy:



Sob o contexto desta amostra sem muita dificuldade pode-se observar dois pontos de concentração, nos fazendo inferir que uma mistura de duas gaussianas (K = 2) é uma modelo razoável. Faremos uma pequena modificação, $\vec{\theta} = (\theta_1, \theta_2) = (1 - \theta, \theta)$desta forma:

$$\pi(y_1, ..., y_N | \mu_1, \mu_2, \sigma_1, \sigma_2, \theta) = \sum_{n=1}^{N} (1 - \theta) \cdot \mathcal{N}(y_n | \mu_1, \sigma_1) + \theta \cdot \mathcal{N}(y_n | \mu_2, \sigma_2)$$

A fim de evitar o problema de discriminação anteriormente mencionado, faremos uma implementação diferente da nossa verossimilhança. Vamos atribuir às duas Gaussianas, os mesmos parâmetros, e uma distribuição Beta simétrica para o peso θ:

$$\mu_1, \mu_2 \sim \mathcal{N}(0, 2), \sigma_1, \sigma_2 \sim \text{Half-}\mathcal{N}(0, 2)$$

$$\theta \sim Beta(5, 5)$$

3 Resultados do Stan

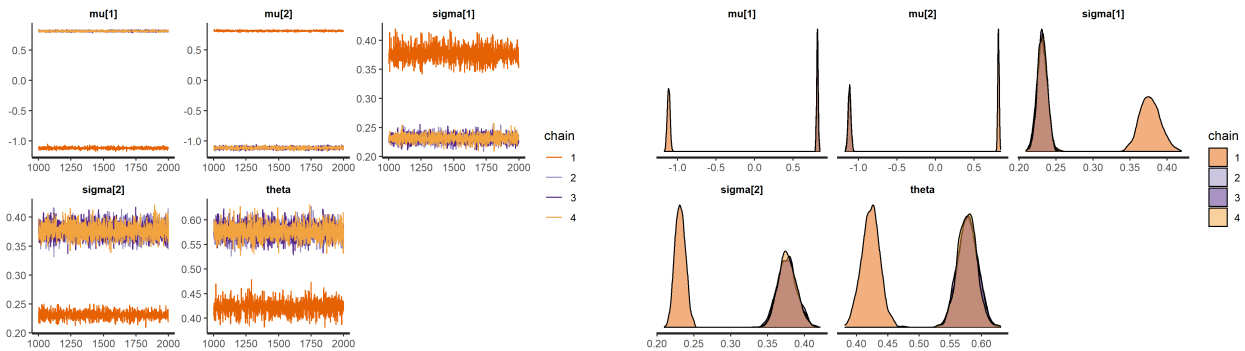
A implementação anteriormente mencionada é passada para o Stan:

```
data {
 int<lower = 0> N;
 vector[N] y;
}

parameters {
  vector[2] mu;
  real<lower=0> sigma[2];
  real<lower=0, upper=1> theta;
}

model {
 sigma ~ normal(0,2);
 mu ~ normal(0,2);
 theta ~ normal(5,5);
 for (n in 1:N)
   target += log_mix(theta,
                     normal_lpdf(y[n] | mu[1], sigma[1]),
                     normal_lpdf(y[n] | mu[2], sigma[2]));
}
```

E fitada aos dados normalizados, por padrão o stan gera 4 cadeias, é um bom número, geralmente o ideal é no mínimo 3, para que não exista riscos elevados de todas elas ficarem presas em mínimos locais. A seguir temos um print do nosso fit:

```
Inference for Stan model: mixture_gaussian.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

             mean se_mean   sd    2.5%     25%     50%     75%   97.5% n_eff   Rhat
mu[1]        0.33    0.59 0.84   -1.14    0.33    0.81    0.82    0.83     2 72.24
mu[2]       -0.63    0.59 0.84   -1.15   -1.12   -1.11   -0.59    0.83     2 54.14
sigma[1]     0.27    0.04 0.06    0.22    0.23    0.23    0.28    0.39     2  7.51
sigma[2]     0.34    0.04 0.06    0.22    0.31    0.37    0.38    0.40     2  5.90
theta        0.54    0.05 0.07    0.40    0.51    0.57    0.58    0.61     2  4.78
lp__      -844.02    0.03 1.55 -847.94 -844.82 -843.70 -842.88 -841.97  2314  1.00
```

Observe os altos valores de Rhat, em situações de convergência ele deveria se aproximar de 1. Isso é um indicativo que as cadeias não estão explorando as mesmas regiões de parâmetros.



Novamente confirmamos isso plotando as cadeias, observe que a primeira cadeia tem resultados bem distoantes das outras 3. Isso mostra a grande relevancia em se simular diversas cadeias em análises via MCMC. Caso não o fizéssemos, não teríamos percebido que ainda precisamos fazer ajustes no modelo.