UFAM

MISTURAS FINITAS DE NORMAIS ASSIMÉTRICAS E DE *t* ASSIMÉTRICAS APLICADAS EM ANÁLISE DISCRIMINANTE

Carina Figueiredo Coelho

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Matemática, da Universidade Federal do Amazonas, como parte dos requisitos necessários à obtenção do título de Mestre em Matemática

Orientador: Prof. Dr. José Raimundo Gomes Pereira

Manaus

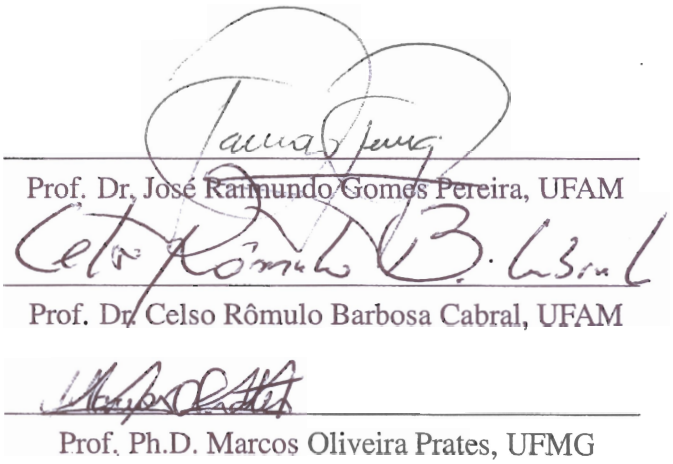
Junho de 2013

MISTURAS FINITAS DE NORMAIS ASSIMÉTRICAS E DE t ASSIMÉTRICAS APLICADAS EM ANÁLISE DISCRIMINANTE

Carina Figueiredo Coelho

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO PROGRAMA DE PÓS-GRADUAÇÃO EM MATEMÁTICA, DA UNIVERSIDADE FEDERAL DO AMAZONAS, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM MATEMÁTICA.

Examinada por:



Prof. Dr. José Raimundo Gomes Pereira, UFAM

Prof. Dr. Celso Rômulo Barbosa Cabral, UFAM

Prof. Ph.D. Marcos Oliveira Prates, UFMG

MANAUS, AM – BRASIL

JUNHO DE 2013

Coelho, Carina Figueiredo

MISTURAS FINITAS DE NORMAIS ASSIMÉTRICAS E DE *t* ASSIMÉTRICAS APLICADAS EM ANÁLISE DISCRIMINANTE/Carina Figueiredo Coelho. – Manaus: UFAM/ICE, 2013.

XII, 14 p.: il.; 29, 7cm.

Orientador: Prof. Dr. José Raimundo Gomes Pereira

Dissertação (mestrado) – UFAM/ICE, Área de Concentração: Estatística, 2013.

Referências Bibliográficas: p. 12 – 14.

1. Análise Discriminante. 2. Mistura Finita de Densidades. 3. Normal Assimétrica. 4. t Assimétrica. I. Pereira, Prof. Dr. José Raimundo Gomes. II. Universidade Federal do Amazonas, UFAM, Área de Concentração: Estatística. III. Título.

Resumo da Dissertação apresentada ao Programa de Pós-Graduação em Matemática, da Universidade Federal do Amazonas, como parte dos requisitos necessários para a obtenção do grau de Mestre em Matemática. (M.Sc.)

MISTURAS FINITAS DE NORMAIS ASSIMÉTRICAS E DE *t* ASSIMÉTRICAS APLICADAS EM ANÁLISE DISCRIMINANTE

Carina Figueiredo Coelho

Junho/2013

Orientador: Prof. Dr. José Raimundo Gomes Pereira

Área de Concentração : Estatística

Investigamos o emprego de misturas finitas de densidades na família normal assimétrica independente, em particular a normal assimétrica e a t assimétrica, para modelar as distribuições condicionais do vetor de características em Análise Discriminante (AD). O objetivo é obter modelos capazes de modelar dados com estruturas mais complexas onde, por exemplo, temos assimetria e multimodalidade, o que muitas vezes ocorrem em problemas reais de AD. Para avaliar esta modelagem, desenvolvemos um estudo de simulação e aplicações em dados reais, analisando a taxa de erro (TE) associadas aos classificadores obtidos com estes modelos de misturas. Foram simulados problemas com diferentes estruturas, relativas à separação e distribuição das classes e o tamanho do conjunto de treinamento. Os resultados do estudo sugerem que os modelos avaliados são capazes de se ajustar aos diferentes problemas estudados, desde os mais simples aos mais complexos, em termos de modelagem das observações para fins de classificação. Com os dados reais, situações onde desconhecemos as formas das distribuições nas classes, os modelos apresentaram TE's razoáveis quando comparados a outros classificadores. Como uma limitação, para os conjuntos de dados analisados, foi observado que a modelagem por misturas finitas necessita de amostras grandes por classe em situações onde a dimensão do vetor de características é relativamente alta.

Palavras-chave: Análise Discriminante, Mistura Finita de Densidades, Normal Assimétrica e t Assimétrica.

Abstract of Dissertation presented to Postgraduate in Mathematics, of the Federal University of Amazonas, as a partial fulfillment of the requirements for the degree of Master of Mathematics. (M.Sc.)

FINITE MIXTURE OF SKEW NORMAL AND SKEW *t* APPLIED IN DISCRIMINANT ANALYSIS

Carina Figueiredo Coelho

June/2013

Advisor: Prof. Dr. José Raimundo Gomes Pereira

Research lines: Statistics

We investigated use of finite mixture models with skew normal independent distributions to model the conditional distributions in discriminat analysis, particularly the skew normal and skew t. To evaluate this model, we developed a simulation study and applications with real data sets, analyzing error rates associated with the classifiers obtained with these mixture models. Problems were simulated with different structures and separations for the classes distributions employing different training set sizes. The results of the study suggest that the models evaluated are able to adjust to different problems studied, from the simplest to the most complex in terms of modeling the observations for classification purposes. With real data, where then shapes distributions of the class is unknown, the models showed reasonable error rates when compared to other classifiers. As a limitation for the analized sets of data was observed that modeling by finite mixtures requires large samples per class when the dimension of the feature vector is relatively high.
keyword: discriminant analysis, finite mixture models, skew normal and skew t.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Neste capítulo apresentamos o problema de Análise Discriminante, seus objetivos e exemplos de aplicações, descrevemos, também, a proposta do trabalho e sua organização

1.1 A Análise Discriminante

O problema abordado em Análise Discriminante(AD) é alocar *objetos* a *classes* previamente definidas. Os objetos podem ser pessoas, plantas, pixels em imagens digitais, etc., e as classes designam as categorias nas quais os objetos devem ser classificados. A AD também é denominada na literatura como *Reconhecimento de Padrões Supervisionado* (RPS) (Hastie *et al.* [10]).

As aplicações de AD ocorrem em diversas áreas de estudos em ciência e tecnologia. Engenharia, biologia, psicologia, medicina, marketing, visão computacional, sensoriamento remoto, inteligência artificial, são alguns exemplos de áreas em que há necessidade de classificar objetos em classes definidas para o problema considerado. Exemplos de algumas situações:

- Detectar tipos de poluição industrial, num espaço geográfico;
- Detectar células anormais em imagens digitais de amostras de sangue;
- Identificar peças defeituosas em um processo de produção por meio de imagens digitais;
- Identificar alvos por meio de sinais de radar;

- Classificar plantas em diferentes espécies;
- Identificar suspeitos de crimes por meio da impressão digital;
- Classificar assinaturas em imagens de documentos como falsas ou verdadeiras;
- Classificação de diferentes tipos de solo em imagens de satélites.

Para as situações expostas acima, que são muitas vezes solucionadas com esforço humano, muitas pesquisas em ciência e tecnologia almejam resolvê-las de maneira mais prática, automatizando tanto quanto possível os procedimentos necessários. Com a disponibilidade cada vez maior de recursos computacionais, muitos destes processos de automatização já são perfeitamente viáveis. (Jain *et al.* [11], Hastie *et al.* [10]).

Nos problemas em AD os objetos são descritos por um conjunto de variáveis, selecionadas de forma que sejam capazes de diferenciar os objetos com relação às classes consideradas no problema a ser resolvido. Este conjunto de variáveis é denominado *vetor de características*, que denotamos por $\mathbf{X}^T = (X_1, ..., X_p)$ que e suas variáveis são denominadas variáveis preditoras, cujas observações são as mensurações feitas sobre o objeto a ser classificado, quantificando assim características discriminantes, ou seja, aspectos relevantes para distinção de classes. Desta forma, uma das etapas do problema em AD consiste em, observado o vetor de características para objeto cuja classe é desconhecida, associá-lo a uma das classes definidas para o problema que seja a mais apropriada ao valor do vetor de características apresentado pelo objeto. Para abordar esta questão, torna-se necessário desenvolver um procedimento que permita implementar esta alocação, denominado de *classificador*.

Na literatura às vezes a AD é apresentada na categoria de problemas de clasificação conjuntamente com Análise de Agrupamento (AA), em ingês *Clustering* (ver, por exemplo Johnson & Wichern [12] e Ripley [24]). Em AA o objetivo é formar grupos com base nas observações do vetor de características e não há classes previamente definidas.

Neste trabalho não abordamos a questão da extração e seleção de variáveis para compor o vetor de características, para esta questão ver, por exemplo, Steel & Louw [25], Maugis *et al.* [18] e Theodoridis & Koutroumbas [26, Cap. 5]. Assumindo que o vetor de características é formado pelas variáveis mais relevantes para o problema, nós abordamos a questão do desenvolvimento do classificador.

1.2 Abordagem Estatística para AD

Para entendermos essa abordagem, segundo Pereira [22], podemos supor a seguinte situação: suponha um problema com M classes e considere uma variável Y que assume valores em $\{1,2,...,M\}$. O valor de Y indica a classe a qual pertence um dado objeto. Pressupomos que existe uma função desconhecida $\mathfrak{F}(.)$ que associa a $\mathbf{X}$ o valor de Y. Um problema consiste em estimar essa função desconhecida ou, ainda, contruir um classificador $r(.)$ que aproxime a função $\mathfrak{F}(.)$. Em AD dispomos de observações do *vetor de características* para objetos em cada uma das classes. A abordagem estatística modela o vetor de características como um vetor aleatório $\mathbf{X}^T = (X_1, X_2, ..., X_p)$, ou seja, suas componentes são consideradas como variáveis aleatórias. Para cada classe, o comportamento de $\mathbf{X}$ é modelado por uma distribuição de probabilidade que é denominada *distribuição condicional da classe*. Nessa modelagem, para cada classe é também incluída a probabilidade do objeto provir da classe, sendo esta probabilidade denominada *probabilidade a priori*. O objetivo é descrever a incerteza inerente à $\mathbf{X}$, procurando caracterizá-lo em termos de seu comportamento mais representativo.

Neste trabalho, empregamos as distribuições condicionais e as probabilidades *a priori* por meio do *Teorema de Bayes* para a obtenção da probabilidade *a posteriori*, que é a probabilidade de um objeto pertencer a uma determinada classe, condicionada ao valor observado do vetor de características $\mathbf{X}$. A metodologia de classificação consiste em alocar o objeto à classe com maior probabilidade *a posteriori*. Um classificador então é construído, isto é, uma regra para classificar futuras observações cujas classes sejam desconhecidas.

1.3 Proposta do Trabalho

As misturas finitas de densidades são extremamente eficientes para modelar distribuições desconhecidas arbitrariamente complexas, inclusive na presença de multimodalidade (McLachlan & Peel [19]). Alguns trabalhos na literatura investigam o emprego de modelos de misturas finitas de densidades para modelar as distribuições em problemas de AD. Dentre estes trabalhos temos Pereira [22] e Fraley & Raftery [8], que empregam misturas finitas de distribuições normais multivariadas, e Andrews & McNicholas [1] que empregam misturas finitas de distribuições t multivariadas.

Alguns trabalhos empregam misturas finitas de distribuições assimétricas, porém, no contexto de AA com uma abordagem denominada *Agrupamento Baseado em Modelos*

(Hastie *et al.* [10]). Wang *et al.* [28], Lee & McLachlan [13] e Lee & McLachlan [14] empregam misturas finitas de distribuições t assimétricas nesse contexto, ou seja, os grupos vão sendo formados, modelados por uma distribuição nesta família para cada um deles.

O objetivo geral é investigar o emprego de misturas finitas de densidades *normais assimétricas* e de *t assimétricas* para modelar as distribuições das classes em problemas de AD. A justificativa para tal é que as misturas finitas, em conjunto com potencial de modelagem das distribuições assimétricas, podem compor um modelo que seja capaz de ajustar satisfatoriamente distribuições que apresentem multimodalidade e assimetria, como é o caso em muitas aplicações de AD.

Como objetivos específicos:

- Revisão de literatura sobre todos os conceitos envolvidos no desenvolvimento desse trabalho;
- Simulação computacional de problemas com especificadas estruturas de distribuições nas classes;
- Aplicações com dados reais da literatura e, em particular, com um problema ainda não abordado com métodos estatísticos.

Na seção seguinte será descrita a organização do trabalho.

1.4 Organização do Trabalho

Esta dissertação está dividida em seis capítulos. Este Capítulo 1 com a Introdução. No Capítulo 2 apresentamos a definição de Mistura Finita de Densidades (MFD), o método de estimação para esse modelo via algoritmo EM. Abordamos ainda algumas particularidades desses modelos, como a identificabilidade de MFD e a estrutura de dados incompletos para o problema de misturas.

No Capítulo 3, apresentamos a definição de Família Normal Assimétrica Independente (NAI), em particular a distribuição normal assimétrica multivariada e a distribuição t assimétrica multivariada, e o processo de estimação via algoritmo EM. Para essas distribuições foi definido o Modelo de Mistura Finita de Distribuições Normais Assimétricas

Independentes (MF-NAI) Multivariadas e a metodologia para a estimação dos parâmetros envolvidos neste modelo também via algoritmo EM.

No Capítulo 4 a abordagem é sobre Análise Discriminante, conceitos e algumas aplicações. Definimos ainda o classificador de Bayes e o classificador de máxima verossimilhança. Nesse contexto, discutimos a abordagem de AD empregando mistura finita de densidades.

No Capítulo 5 são apresentados os resultados obtidos por meio dos estudos de simulações e aplicações com dados reais da literatura e com dados reais originais, com os quais ainda não tinha sido empregada uma metodologia estatística de classificação.

As conclusões e sugestões são apresentadas no Capítulo 6.

No Apêndice A são apresentados os programas implementados neste trabalho.

Capítulo 2

Mistura Finita de Densidades

Neste capítulo apresentamos a definição de mistura finita de densidades e suas principais particularidades. Discutimos a estimação dos seus parâmetros por máxima verossimilhança, via algoritmo EM (do termo em inglês, *Expectation and Maximization*), os resultados pertinentes a esta abordagem e o critério de seleção de modelos BIC (do termo em inglês *Bayesian Information Criterion*), adotado no trabalho.

2.1 O Modelo de Mistura Finita de Densidades

Um vetor aleatório $\mathbf{X} = (X_1, X_2, ..., X_p)^T$ tem distribuição dada por uma *mistura finita de densidades* (MFD), com g componentes, se sua densidade é da forma

$$f(\mathbf{x}; \boldsymbol{\Theta}) = \sum_{j=1}^{g} p_j f_j(\mathbf{x}; \boldsymbol{\theta}_j), \tag{2.1}$$

onde $p_j \geq 0$, $j = 1, ..., g$, com $\sum_{j=1}^{g} p_j = 1$, são chamados *pesos da mistura*, a densidade $f_j(.; \theta_j)$ é denominada a j-ésima *componente da mistura*, indexada por um vetor de parâmetros $\boldsymbol{\theta}_j$ e $\boldsymbol{\Theta} = ((p_1, ..., p_g)^T, \boldsymbol{\theta}_1^T, ..., \boldsymbol{\theta}_g^T)^T$. A Figura 1.1 apresenta um conjunto de dados ajustados por uma MFD normais univariadas, com $g = 3$. Observe que o conjunto de dados apresenta uma distribuição complexa, com multimodalidade, situação em que uma única família paramétrica de distribuições não produziria uma modelagem satisfatória.

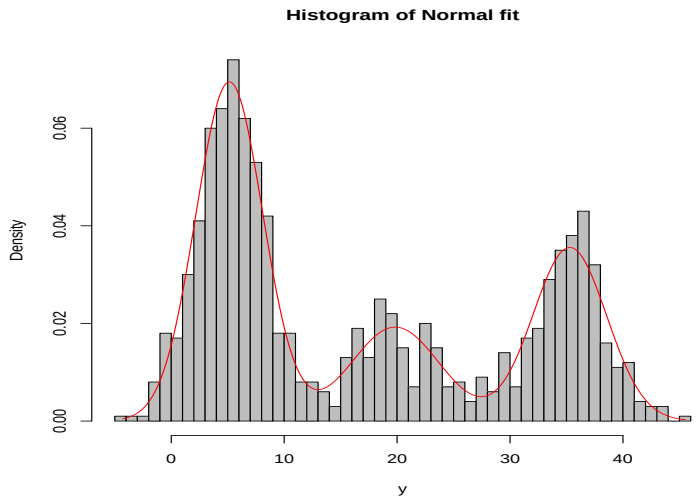


Figura 2.1: Dados ajustados por uma MFD normais univariadas, com $g = 3$.

A Figura 1.2 apresenta um conjunto de dados ajustados por uma MFD normais bivariadas, com $g = 2$. Observe que este conjunto de dados apresenta também uma distribuição complexa, com multimodalidade.

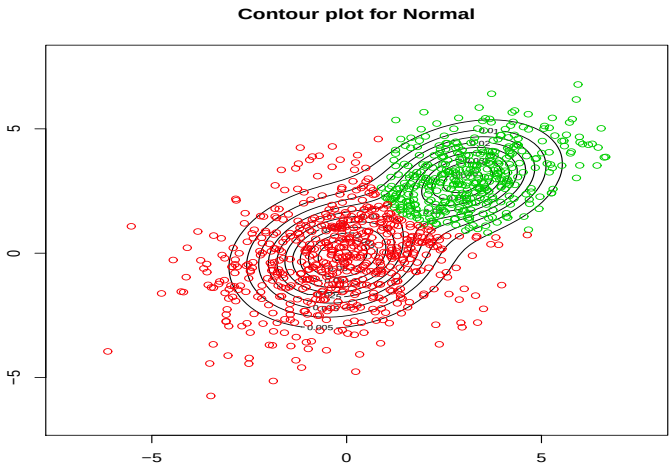


Figura 2.2: Dados ajustados por uma MFD normais bivariadas, com $g = 2$.

Para o emprego do modelo definido em (1.1), é necessário estimar os parâmetros em $\boldsymbol{\Theta}$ que, na maioria das aplicações, é implementado pelo método de máxima verossimilhança (ver, por exemplo, McLachlan & Peel [19] e Lee & McLachlan [13]). No caso de MFD a estimação de por máxima verossimilhança exige o emprego de métodos numéricos de maximização e o mais comumente empregado é o algoritmo EM (de Expectation and Maximization algorithm) (Dempster *et al.* [6]). Na definição apresentada, o número de componentes g está fixado, no entanto, na prática seu valor é desconhecido sendo ne-

cessário estimá-lo a partir de dados observados para $\mathbf{X}$. Esta questão de estimar g pode ser abordada no contexto de *Seleção de Modelos*, com o emprego de uma função-critério cuja otimização indique o número de componentes adequado aos dados.

2.1.1 Identificabilidade de MFD

Segundo McLachlan & Peel [19] a estimação de um parâmetro $\boldsymbol{\Theta}$, para uma distribuição $f(\mathbf{x}, \boldsymbol{\Theta})$, baseada nas observações de $\mathbf{X}$, só faz sentido se $\boldsymbol{\Theta}$ é identificável, isto é, possuir uma caracterização única a partir de seus parâmetros. Em geral, uma família paramétrica de funções densidades de probabilidades $f(.;\boldsymbol{\Theta})$ é dita ser identificável se valores distintos de $\boldsymbol{\Theta}$ determinam membros distintos da família de densidades. Para ver isso, considere a família de distribuição

$$\{f(\cdot;\boldsymbol{\Theta}) : \boldsymbol{\Theta} \in \Omega\},$$

onde Ω é o espaço paramétrico especificado. Então

$$f(\cdot;\boldsymbol{\Theta}) = f(\cdot;\boldsymbol{\Theta}')$$

se, e somente se,

$$\boldsymbol{\Theta} = \boldsymbol{\Theta}'.$$

A identificabilidade no caso de mistura de densidades, apresenta uma característica mais específica. Como exemplo, considere uma mistura de duas densidades normais univariadas com médias μ_1 e μ_2, e variâncias iguais a 1. Temos

$$f(x;\boldsymbol{\Theta}) = \frac{p_1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x-\mu_1)^2\right] + \frac{p_2}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x-\mu_2)^2\right].$$

onde $\boldsymbol{\Theta} = ((p_1, p_2)^T, \mu_1^T, \mu_2^T)^T$. Note que, com os valores dos parâmetros fixados, se permutarmos os índices em $\boldsymbol{\Theta}$ a densidade $f(.|\boldsymbol{\Theta})$ terá o mesmo valor em cada $x \in \mathbb{R}$. Portanto, para a identificabilidade no contexto de misturas é necessário mais uma condição, a de *permutabilidade*.

Seja $\mathscr{F} = \{\psi(\mathbf{x};\boldsymbol{\theta}) : \boldsymbol{\theta} \in \Omega, \mathbf{x} \in \mathbb{R}^p\}$ uma família paramétrica de densidades e

$$\begin{aligned}\mathscr{P} &= \{f(\mathbf{x};\Theta) : f(\mathbf{x};\Theta) = \sum_{j=1}^{g} p_j \psi(\mathbf{x};\boldsymbol{\theta}_j), \quad p_j \geq 0, \\ &\quad \sum_{j=1}^{g} p_j = 1, \psi(\mathbf{x};\boldsymbol{\theta}) \in \mathscr{F}, \quad \Theta = ((p_1,...,p_g)^T, \boldsymbol{\theta}_1^T, ..., \boldsymbol{\theta}_g^T)^T\}. \end{aligned} \tag{2.2}$$

uma família de MFD. A classe $\mathscr{P}$ é dita identificável se, para quaisquer dois membros

$$f(\mathbf{x};\Theta) = \sum_{j=1}^{g} p_j \psi(\mathbf{x};\boldsymbol{\theta}_j) \quad \text{e} \quad f(\mathbf{x};\Theta') = \sum_{j=1}^{g'} p'_j \psi(\mathbf{x};\boldsymbol{\theta}'_j),$$

tem-se que $f(\mathbf{x};\Theta) = f(\mathbf{x};\Theta')$ se, e somente se, $g = g'$ e ainda podemos permutar os índices das componentes de forma que $p_j = p'_j$ e $\psi(\mathbf{x};\boldsymbol{\theta}_j) = \psi(\mathbf{x};\boldsymbol{\theta}'_j)$, com $j = 1,...,g$.

Como observado em McLachlan & Peel [19, Seção. 1.4], a falta de identificabilidade não é preocupante na estimação de máxima verossimilhança via algoritmo EM, no entanto, no contexto de estimação bayesiana pode causar problemas. Também, isto não se constitui um problema em aplicações onde o interesse principal é estimar o valor da densidade em observações específicas que, como será visto, é o caso em Análise Discriminante.

2.2 Estimação por Máxima Verossimilhança

Nesta seção descrevemos a estimação por máxima verossimilhança e discutimos algumas particularidades no contexto de MFD.

Definição 2.1. *Seja* $\mathbf{X}$ *um vetor aleatório com distribuição dada por* $f(\cdot;\Theta)$*, com* $\Theta \in \Omega$*. Dado* $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ *um conjunto de n observações independentes de* $\mathbf{X}$*, a Função de Verossimilhança (FV) é definida como*

$$L(\Theta) = L(\Theta|\mathbf{x}) = \prod_{i=1}^{n} f(\mathbf{x}_i;\Theta). \tag{2.3}$$

Sendo $f(\cdot;\Theta)$ uma mistura finita de densidades, temos

$$L(\Theta) = \prod_{i=1}^{n} \{\Sigma_{j=1}^{g} p_j f_j(\mathbf{x}_i; \boldsymbol{\theta}_j)\}, \tag{2.4}$$

a função de verossimilhança para o modelo de MFD.

Definição 2.2. *Se existe um único valor* $\widehat{\Theta} \in \Omega$ *que maximiza a Função de Verossimilhança, então* $\widehat{\Theta}$ *é denominado* Estimador de Máxima Verossimilhança *(EMV) de* Θ.

Devido ao problema da falta de identificabilidade, existem situações em que a FV atinge máximo local para diferentes valores de Θ, por esse motivo, as equações de verossimilhança tem várias raízes. Como observado em McLachlan & Krishnan [21], esse problema pode ser grave quando há interesse na estimação dos valores específicos para os parâmetros envolvidos. Por outro lado, se o objetivo é apenas estimar o valor da densidade em pontos de interesse que, como já mencionado, é o caso em Análise Discriminante, isto não se constitui em problema (ver Ripley [24, Cap. 6]).

Outra situação particular é que no modelo de misturas finitas a FV pode não ser limitada superiormente (ver McLachlan & Peel [19, Seção 1.13]), desta forma, o EMV pode não existir. Como observado em Ripley [24, Seção 6.4], em muitos problemas reais, por exemplo, nas aplicações em Análise Discriminante, é necessário somente obter uma "boa aproximação" para o valor estimado da densidade nos pontos de interesse, portanto, é possível obter uma estimativa para Θ que permita o emprego das MFD como modelos para aproximarem as distribuições envolvidas no problema.

Na prática, a determinação dos EMV para MFD envolvem várias dificuldades devido à complexidade da dependência da FV nos parâmetros. As equações de máxima verossimilhança não são lineares, não sendo possível obter soluções analíticas, portanto, é necessário o emprego de métodos numéricos de maximização de funções (Pereira [22]). Para esse fim, segundo McLachlan & Krishnan [21] e Lee & McLachlan [14], o algoritmo EM é o mais empregado nesse contexto.

2.3 Algoritmo EM para Modelos de Misturas Finitas

Nesta seção descrevemos o algoritmo EM para mistura finita de densidades, sendo necessário definirmos inicialmente a questão da estrutura de dados incompletos que é empregada pelo algoritmo.

Seja $\mathbf{X}_1, ...\mathbf{X}_n$ uma amostra aleatória de $\mathbf{X}$, com distribuição dada por (1.1). Para $i = 1, ..., n$, seja $\mathbf{Z}_i$ um vetor aleatório, com $\mathbf{Z}_i = (Z_{i1}, ..., Z_{ig})$, cujas componentes são definidas como variáveis indicadoras da forma

$$Z_{ij} = \begin{cases} 1, & \text{se } \mathbf{X}_i \sim f_j(\mathbf{x}; \boldsymbol{\theta}_j); \\ 0, & \text{se } \mathbf{X}_i \sim f_l(\mathbf{x}; \boldsymbol{\theta}_l),\ l \neq j. \end{cases}$$

Podemos interpretar essa variável Z_{ij} como uma variável latente, não observável, associada ao vetor $\mathbf{X}_i$ e indicando qual componente da mistura descreve sua distribuição.

Neste contexto, os valores observados $\mathbf{x}_1, ..., \mathbf{x}_n$ de $\mathbf{X}_1, ...\mathbf{X}_n$, são considerados *dados incompletos*. Denotando por $\mathbf{z}_1, ...\mathbf{z}_n$ os valores para $\mathbf{Z}_1, ...\mathbf{Z}_n$, os vetores de *dados completos* são, portanto,

$$\mathbf{y}_i = (\mathbf{x}_i, \mathbf{z}_i), \quad i = 1, ..., n.$$

Assumimos então que o vetor $\mathbf{Z}_i$ tem distribuição multinomial, considerando uma retirada em g categorias, com probabilidades $p_1, ..., p_g$, isto é, formam uma amostra aleatória

$$\mathbf{Z}_1, ..., \mathbf{Z}_n \sim Multi_g(1; p_1, ..., p_g),$$

portanto a distribuição de $\mathbf{Z}_i$ é da forma

$$f(\mathbf{z}_i; \boldsymbol{\alpha}) = \prod_{j=1}^{g} p_j^{z_{ij}}, \text{ onde } \boldsymbol{\alpha} = (p_1, ..., p_g). \tag{2.5}$$

Pela construção acima, o j-ésimo peso da mistura (p_j) pode ser interpretado como a probabilidade *a priori* da i-ésima observação provir da j-ésima componente da mistura, com $i = 1, ..., n$.

Pelas suposições estabelecidas, a distribuição conjunta para os dados completos $\mathbf{y}_i$ é da forma

$$
\begin{aligned}
f(\mathbf{y}_i;\boldsymbol{\Theta}) &= f(\mathbf{x}_i|\mathbf{z}_i;\boldsymbol{\Theta})\, f(\mathbf{z}_i;\boldsymbol{\alpha}) \\
&= \prod_{j=1}^{g} f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)^{z_{ij}}\, f(\mathbf{z}_i;\boldsymbol{\alpha}) \\
&= \prod_{j=1}^{g} f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)^{z_{ij}} p_j^{z_{ij}}. \qquad (2.6)
\end{aligned}
$$

Para o desenvolvimento do EM, construimos a FV dos dados completos $\mathbf{y} = (\mathbf{y}_1, ..., \mathbf{y}_n)$, dada por

$$
L(\boldsymbol{\Theta}) = L(\boldsymbol{\Theta}|\mathbf{y}) = \prod_{i=1}^{n} f(\mathbf{y}_i;\boldsymbol{\Theta}) = \prod_{i=1}^{n}\prod_{j=1}^{g} f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)^{z_{ij}} p_j^{z_{ij}}, \qquad (2.7)
$$

e o seu logaritmo (na base e) dado por

$$
\begin{aligned}
l_c(\boldsymbol{\Theta}) &= \log L(\boldsymbol{\Theta}) = \sum_{i=1}^{n}\sum_{j=1}^{g} z_{ij}\{\log p_j + \log f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)\} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{g} z_{ij}\log p_j + \sum_{i=1}^{n}\sum_{j=1}^{g} z_{ij}\log f_j(\mathbf{x}_i;\boldsymbol{\theta}_j). \qquad (2.8)
\end{aligned}
$$

Para a construção do algoritmo EM determinamos a função Q

$$
Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) = E\{l_c(\boldsymbol{\Theta})|\mathbf{x};\boldsymbol{\Theta}^{(k)}\}, \qquad (2.9)
$$

onde $\mathbf{x} = (\mathbf{x}_1, ..., \mathbf{x}_n)$ e a notação (k) sobrescrito informa que essa é a estimação do parâmetro obtida no instante k do algoritmo. Observe que $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)})$ é a esperança calculada usando $\boldsymbol{\theta}^{(k)}$ como valor para $\boldsymbol{\theta}$. O algoritmo, então, se desenvolve em dois passos:

1. Passo E: Determinar $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)})$;
2. Passo M: Escolher $\boldsymbol{\Theta}^{(k+1)}$ maximizando $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)})$.

Uma propriedade importante do algoritmo EM é que a forma como as estimativas $\boldsymbol{\Theta}^{(k)}$ são determinadas garante que $Q(\boldsymbol{\Theta}^{(k+1)}|\boldsymbol{\Theta}^{(k)}) \geq Q(\boldsymbol{\Theta}^{(k)}|\boldsymbol{\Theta}^{(k)})$ (ver, por exemplo

Dempster *et al.* [6]). É demonstrado (ver, por exemplo, Pereira [22]) que este fato implica em

$$l(\boldsymbol{\Theta}^{(k+1)}) \geq l(\boldsymbol{\Theta}^{(k)}), \tag{2.10}$$

portanto, as aproximações $l(\boldsymbol{\Theta}^{(k)})$ obtidas pelo algoritmo EM geram uma sequência $\{l(\boldsymbol{\Theta}^{(k)})\}$ não-decrescente.

Na implementação do algoritmo, os passos E e M são repetidos alternadamente até que um critério de convergência adequado seja atingido. Várias propostas para critérios de parada são discutidas na literatura (ver, por exemplo, McLachlan & Peel [19] e McLachlan & Krishnan [21]). Neste trabalho, o critério de convergência adotado é da forma

$$\left| \frac{l(\boldsymbol{\Theta}^{(k+1)})}{l(\boldsymbol{\Theta}^{(k)})} - 1 \right| < c, \tag{2.11}$$

com c suficientemente pequeno.

Para a expressão (1.9), obtemos

$$Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) = \sum_{i=1}^{n}\sum_{j=1}^{g} E[Z_{ij}|\mathbf{x};\boldsymbol{\Theta}^{(k)}] \ln p_j + \sum_{i=1}^{n}\sum_{j=1}^{g} E[Z_{ij}|\mathbf{x};\boldsymbol{\Theta}^{(k)}] \ln f_j(\mathbf{x}_i;\boldsymbol{\theta}_j^{(k)}), \tag{2.12}$$

onde $\boldsymbol{\Theta}^{(k)} = ((p_1^{(k)},...,p_g^{(k)})^T, \boldsymbol{\theta}_1^{T(k)},...,\boldsymbol{\theta}_g^{T(k)})^T$.

De (1.12), vemos que é necessário determinar

$$\tau_{ij}^{(k)} \stackrel{\text{def}}{=} E[Z_{ij}|\mathbf{x};\boldsymbol{\Theta}^{(k)}] = P[Z_{ij}=1|\mathbf{x};\boldsymbol{\Theta}^{(k)}] = P[Z_{ij}=1|\mathbf{x}_i,\boldsymbol{\Theta}^{(k)}], \tag{2.13}$$

Considerando a distribuição dada em (1.5), temos que

$$P[Z_{ij}=1|\boldsymbol{\Theta}^{(k)}] = P[Z_{ij}=1, Z_{il}=0 \forall l \neq j|\boldsymbol{\Theta}^{(k)}] = p_j^{(k)}. \tag{2.14}$$

Usando (1.6), vemos que

$$f(\mathbf{x}_i|\mathbf{Z}_i,\boldsymbol{\Theta}^{(k)}) = \prod_{j=1}^{g} (f_j(\mathbf{x}_i;\boldsymbol{\theta}_j^{(k)}))^{z_{ij}}, \tag{2.15}$$

e, portanto, temos que

$$f(\mathbf{x}_i|Z_{ij}=1;\boldsymbol{\Theta}^{(k)}) = f_j(\mathbf{x}_i;\boldsymbol{\theta}_j^{(k)}). \tag{2.16}$$

Empregando o Teorema de Bayes, vemos que

$$\tau_{ij}^{(k)} = P[Z_{ij}=1|\mathbf{x}_i,\boldsymbol{\Theta}^{(k)}] = \frac{P[Z_{ij}=1|\boldsymbol{\Theta}^{(k)}]\, f(\mathbf{x}_i|Z_{ij}=1,\boldsymbol{\Theta}^{(k)})}{f(\mathbf{x}_i;\boldsymbol{\Theta}^{(k)})}. \tag{2.17}$$

Usando (1.1), (1.14) e (1.16) em (1.17), obtemos

$$\tau_{ij}^{(k)} = \frac{p_j^{(k)} f_j(\mathbf{x}_i;\boldsymbol{\theta}_j^{(k)})}{\sum_{t=1}^{g} p_t^{(k)} f_t(\mathbf{x}_i;\boldsymbol{\theta}_t^{(k)})}. \tag{2.18}$$

Da expressão acima, vemos que $\tau_{ij}^{(k)}$ é uma estimativa da probabilidade da observação $\mathbf{x}_i$ provir da componente $f_j(\cdot;\boldsymbol{\theta}_j^{(k)})$ da mistura, com base em uma dada estimativa $\boldsymbol{\Theta}^{(k)}$ do vetor de parâmetros $\boldsymbol{\Theta}$.

Agora, usando (1.18) em (1.12), podemos escrever

$$\begin{aligned} Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) &= \sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(k)} \log p_j + \sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(k)} \log f_j(\mathbf{x}_i;\boldsymbol{\theta}_j) \\ &= Q_1(\boldsymbol{\alpha}) + Q_2(\boldsymbol{\theta}), \end{aligned} \tag{2.19}$$

onde $\boldsymbol{\alpha} = (p_1,...,p_g)$ e $\boldsymbol{\theta} = (\boldsymbol{\theta}_1,...,\boldsymbol{\theta}_g)$,

$$Q_1(\boldsymbol{\alpha}) = \sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(g)} \log p_j \tag{2.20}$$

e

$$Q_2(\boldsymbol{\theta}) = \sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(k)} \log f_j(\mathbf{x}_i;\boldsymbol{\theta}_j). \tag{2.21}$$

Como pode ser visto na expressão (1.19), o problema de maximização de $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)})$ é considerado, separadamente, para $\boldsymbol{\alpha}$ e $\boldsymbol{\theta}$, ou seja, dois problemas que envolvem

$$\frac{\partial Q_1(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}} = \mathbf{0} \quad \text{e} \quad \frac{\partial Q_2(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0}.$$

Note que o primeiro problema de maximização tem uma solução única que é explicitamente determinada, independente da forma funcional das componentes da mistura. Considerando as restrições sobre $\boldsymbol{\alpha}$, os p_j são não negativos e somam 1, a solução é obtida com o emprego de multiplicadores de Lagrange, sendo dada por (ver Dempster *et al.* [6])

$$p_j^{(k+1)} = \frac{1}{n}\sum_{i=1}^{n} \tau_{ij}^{(k)}, \qquad j = 1,2,...,g. \tag{2.22}$$

Dessa maneira, determinamos $\tau_{ij}^{(k)}$ por meio de (1.18) e as aproximações para $\boldsymbol{\alpha}^{(k+1)}$ por (1.22).

No segundo problema de maximização, com relação à $\boldsymbol{\theta}$, vemos que a solução depende da forma funcional das componentes $f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)$, pois devemos solucionar

$$\sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(k)} \frac{\partial}{\partial \boldsymbol{\theta}} \ln f_j(\mathbf{x}_i;\boldsymbol{\theta}_j) = \mathbf{0}. \tag{2.23}$$

Para a equação (1.23) não há garantia de solução única e bem definida. Isto será discutido no Capítulo 3, no contexto de mistura finita de densidades com as componentes na família de *distribuições normais assimétricas independentes*.

2.4 Estimação do número de componentes

A abordagem da estimação do número de componentes consiste em considerar uma função-critério cuja otimização indique o número de componentes do modelo adequado aos dados. Uma função-critério, em geral, seleciona o número $\widehat{g}$ de componentes da forma

$$\widehat{g} = arg\,min_g\{C(\widehat{\boldsymbol{\Theta}}_{(g)}), g = g_{min},...,g_{max}\},$$

onde $C(\widehat{\boldsymbol{\Theta}}_{(g)})$ é o valor da função-critério para o modelo estimado com dimensão g, no caso de mistura finita de densidades, o modelo com g componentes. Existem várias propostas na literatura para a função-critério (ver, por exemplo, McLachlan & Peel [19] e Hastie *et al.* [10]). Segundo Andrews & McNicholas [1], um dos critérios de seleção de modelos mais empregado é o Critério de Informação Bayesiano (ver, também, Wang & Hu [27], Wang *et al.* [28] e Fraley & Raftery [8]), que denotaremos por BIC, devido ao termo em inglês *Bayesian Information Criterion*, que será adotado neste trabalho.

O BIC é baseado na teoria Bayesiana de seleção de modelos, onde são considerados vários possíveis modelos, com suas probabilidades *a priori*, objetivando selecionar

o modelo com a maior probabilidade "a posteriori", dadas as observações. Considere os modelos $M_1,...,M_G$ e suas respectivas probabilidades *a priori* $P(M_g), g = 1,2,3,...,G$. Pelo Teorema de Bayes, "a posteriori" de M_g dado $\mathbf{x} = (\mathbf{x}_1,...,\mathbf{x}_n)$ é

$$P(M_g|\mathbf{x}) = \frac{f(\mathbf{x}|M_g)P(M_g)}{\sum_{t=1}^{G} f(\mathbf{x}|M_t)P(M_t)}. \tag{2.24}$$

Para essa expressão, é necessário determinar $f(\mathbf{x}|M_g)$, que denota a distribuição de $\mathbf{x}$ segundo o modelo M_g. Como os parâmetros são geralmente desconhecidos, obtemos (1.24) integrando sobre o espaço paramétrico Ω,

$$f(\mathbf{x}|M_g) = \int L(\mathbf{x}|\boldsymbol{\Theta}, M_g)\pi(\boldsymbol{\Theta}_{(g)}|M_g)d\boldsymbol{\Theta}_{(g)}, \tag{2.25}$$

onde $\pi(\boldsymbol{\Theta}_{(g)}|M_g)$ é a distribuição *a priori* para $\boldsymbol{\Theta}_{(g)}$ e $L(\mathbf{x}|\boldsymbol{\Theta}, M_g)$ é a função de verossimilhança para o modelo M_g com vetor de parâmetros $\boldsymbol{\Theta}_{(g)}$.

Se as probabilidades $P(M_g)$ em (1.24) forem iguais, o procedimento seleciona o modelo que apresentar maior valor para $f(\mathbf{x}|M_g)$, denominada de *verossimilhança integrada*. Em todos os casos, é necessário obter uma aproximação para a integral (1.25), o desenvolvimento para obter esta aproximação está descrito em Basso [4], sendo o resultado da forma

$$BIC(g) = -2l(\widehat{\boldsymbol{\Theta}}_g) + \upsilon(g)\log n, \tag{2.26}$$

onde $l(\widehat{\boldsymbol{\Theta}}_{(k)})$ é o logaritmo da função de máxima verossimilhança para o modelo com vetor de parâmetros estimados $\widehat{\boldsymbol{\Theta}}_{(g)}$, e $\upsilon(g)$ é o número de parâmetros livres no modelo de dimensão g. Portanto, o procedimento seleciona o modelo com o número de componentes que apresenta o menor valor para o BIC.

Na literatura, é mencionado que o BIC assintoticamente tende a selecionar o modelo de dimensão correta, por isso é denominado *consistente em ordem* (ver referências em Pereira [22]). Apesar de não ter sido desenvolvido para modelos de misturas, na prática este critério tem apresentado resultados satisfatórios para selecionar o número de componentes da mistura ao aproximar densidades desconhecidas empregando misturas finitas de densidades (ver Pereira [22], Basso [4] e as referências nestes trabalhos).

Capítulo 3

Mistura Finita de Normais Assimétricas Independentes

Neste capítulo definimos a Família Normal Assimétrica Independente, abordamos algumas de suas propriedades e duas distribuições particulares nesta família de interesse neste trabalho, a saber, a distribuição normal assimétrica e a distribuição t assimétrica multivariadas. Apresentamos a mistura finita de densidades nesta família e, também, a estimação dos parâmetro destes modelos, via um algoritmo do tipo EM.

3.1 Família Normal Assimétrica Independente

Para definirmos a familia normal assimétrica independente (NAI), originalmente em inglês *skew normal independent distributions (SNI)*, inicialmente iremos definir uma distribuição em particular, a normal assimétrica (NA) multivariada, na versão empregada em Cabral *et al.* [5] (para algumas versões ver Azzalini [3] e Arellano-Vale & Azzalini [2]).

Um vetor $\mathbf{X}_{(p\times 1)}$ é dito ter distribuição *normal assimétrica multivariada*, com vetor de locação $\boldsymbol{\mu}$, matriz de dispersão positiva definida $\boldsymbol{\Sigma}$ e vetor de assimetria $\boldsymbol{\lambda}$, se sua densidade é da forma

$$NA_p(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma},\ \boldsymbol{\lambda}) = 2\phi_p(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma})\Phi(\ \boldsymbol{\lambda}^T\boldsymbol{\Sigma}^{-1/2}(\mathbf{x}-\boldsymbol{\mu})). \tag{3.1}$$

onde $\boldsymbol{\lambda}^T$ denota o transposto de $\boldsymbol{\lambda}$, $\phi_p(\cdot|\boldsymbol{\mu},\boldsymbol{\Sigma})$ é a densidade da distribuição normal p-variada com vetor de médias $\boldsymbol{\mu}$ e matriz de covariâncias $\boldsymbol{\Sigma}$, $N_p(\boldsymbol{\mu},\boldsymbol{\Sigma})$, e $\Phi(\cdot)$ é a função de distribuição da normal padrão univariada, $N(0,1)$. A matriz $\boldsymbol{\Sigma}^{-1/2}$ é a matriz raiz

quadrada de $\boldsymbol{\Sigma}$, isto é, uma matriz simétrica que satisfaz $\boldsymbol{\Sigma}^{-1/2}\boldsymbol{\Sigma}^{-1/2} = \boldsymbol{\Sigma}^{-1}$.

Para a distribuição em (**??**) empregamos a notação $\mathbf{X} \sim NA_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda})$ e observe que para $\boldsymbol{\lambda} = \mathbf{0}$, temos $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Na Figura **??** ilustramos o comportamento dessa distribuição para o caso univariado com diferentes valores para λ.

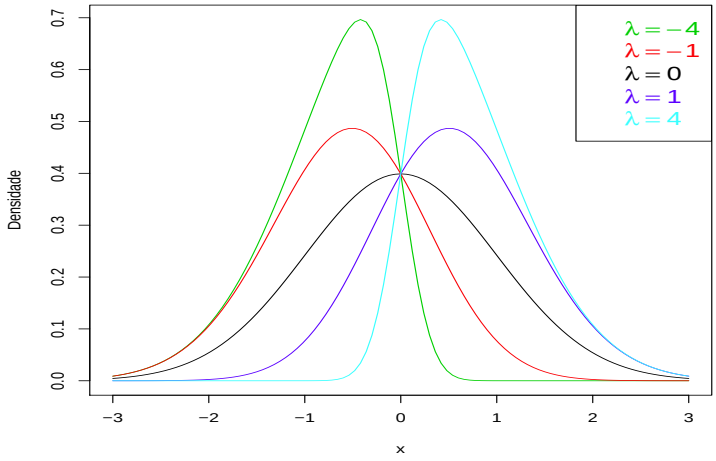


Figura 3.1: Gráfico da densidade $NA(0, 1, \lambda)$.

É importante ressaltar que a distribuição NA, devido ao parâmetro $\boldsymbol{\lambda}$, é adequada para modelar dados com assimetria em sua distribuição, diferentemente da distribuição normal usual. Na Figura **??**, temos um exemplo do comportamento dessa distribuição no caso bivariado.

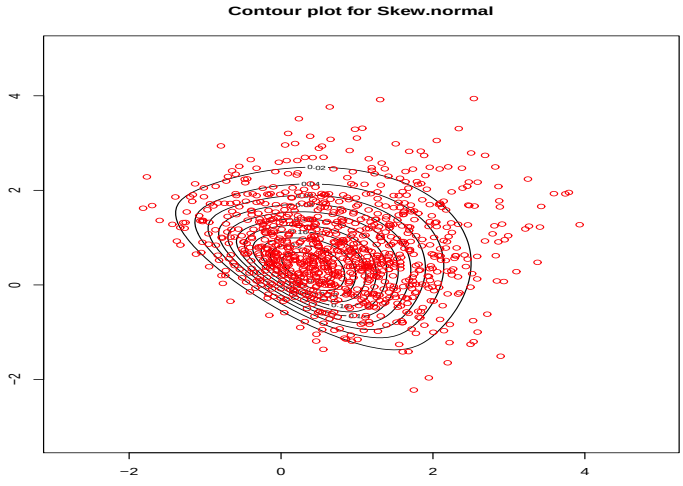


Figura 3.2: Dados ajustados por uma NA bivariada.

Definição 3.1. *Um vetor* $\mathbf{X}$ *tem distribuição pertencente à Família Normal Assimétrica Independente (NAI) quando* $\mathbf{X} = \boldsymbol{\mu} + U^{-1/2}\mathbf{Z}$*, onde* $\boldsymbol{\mu}$ *é o vetor de locação p-dimensional,* $\mathbf{Z} \sim NA_p(\mathbf{0}, \boldsymbol{\Sigma}, \boldsymbol{\lambda})$ *e* U *é uma variável aleatória positiva, independente de* $\mathbf{Z}$*, com função de distribuição* $H(.|\boldsymbol{\nu})$ *e densidade* $h(.|\boldsymbol{\nu})$ *(onde* $\boldsymbol{\nu}$ *é o parâmetro da função distribuição* H*, sendo um escalar ou vetor).*

Para a distribuição definida acima, empregamos a notação $\mathbf{X} \sim NAI_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, H)$. Temos que $\mathbf{X}|U = u \sim NA_p(\boldsymbol{\mu}, u^{-1}\boldsymbol{\Sigma}, \boldsymbol{\lambda})$, o que significa que as distribuições NAI são misturas escalonadas de distribuições NA.

Escrevendo a distribuição conjunta de $(\mathbf{X}, U)$ em termos do produto da distribuição $\mathbf{X}|U = u$ pela distribuição de U, então a distribuição marginal de X é dada por:

$$NAI_p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, H) = 2\int_0^\infty \phi_p(\mathbf{x}|\boldsymbol{\mu}, u^{-1}\boldsymbol{\Sigma})\Phi(u^{1/2}\,\boldsymbol{\lambda}^T\boldsymbol{\Sigma}^{-1/2}(\mathbf{x}-\boldsymbol{\mu}))dH(u|\boldsymbol{\nu}). \tag{3.2}$$

Computacionalmente, as expressões das distribuições pertencentes à família NAI não são de fácil implementação para fins de estimação. Podemos reparametrizá-las, de tal maneira que os resultados teóricos não sejam comprometidos. Como mencionado em Cabral *et al.* [5], uma reparametrização usual, e válida para toda distribuição pertencente à família NAI, é dada por

$$\boldsymbol{\Delta} = \boldsymbol{\Sigma}^{1/2}\boldsymbol{\delta}, \quad \boldsymbol{\Gamma} = \boldsymbol{\Sigma}^{1/2}(\mathbf{I} - \boldsymbol{\delta}\boldsymbol{\delta}^T)\boldsymbol{\Sigma}^{1/2} = \boldsymbol{\Sigma} - \boldsymbol{\Delta}\boldsymbol{\Delta}^T, \tag{3.3}$$

onde $\mathbf{I}$ denota a matriz identidade de dimensão apropriada e $\boldsymbol{\delta} = \dfrac{\boldsymbol{\lambda}}{\sqrt{1+\boldsymbol{\lambda}^T\boldsymbol{\lambda}}}$.

A reparametrização em (**??**) é uma função "um a um", assim a recuperação dos parâmetros originais é dada pelas expressões:

$$\boldsymbol{\lambda} = \frac{(\boldsymbol{\Gamma} + \boldsymbol{\Delta}\boldsymbol{\Delta}^T)^{-1/2}\boldsymbol{\Delta}}{[1 - \boldsymbol{\Delta}^T(\boldsymbol{\Gamma} + \boldsymbol{\Delta}\boldsymbol{\Delta}^T)^{-1}\boldsymbol{\Delta}]^{1/2}}, \quad \boldsymbol{\Sigma} = \boldsymbol{\Gamma} + \boldsymbol{\Delta}\boldsymbol{\Delta}^T. \tag{3.4}$$

A implementação tem por base duas proposições que apresentamos a seguir. A notação empregada $NT(a, b^2, B)$ representa a distribuição normal truncada num conjunto B, com os parâmetros indicados antes do truncamento, $NI(\cdot,\cdot,\cdot)$ uma distribuição na família *normal independente* com os parâmetros indicados (ver, Cabral *et al.* [5], Seção 2.1) e $\stackrel{d}{=}$ significa "tem a mesma distribuição de".

Proposição 3.1. $\mathbf{X} \sim NAI_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, H)$ *admite a seguinte representação hierárquica:*

$$\mathbf{X}|U = u, T = t \sim N_p(\boldsymbol{\mu} + \boldsymbol{\Delta} t, u^{-1}\Gamma), \quad T|U = u \sim NT(0, u^{-1}, (0, \infty)), \quad U \sim H(u|\boldsymbol{\nu}).$$

Proposição 3.2. *Sejam* $\mathbf{X} \sim NAI_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, H)$, $\mathbf{X}_0 \sim NI_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}, H)$. *Defina* $U_x \stackrel{d}{=} (U|\mathbf{X}_0 = x)$. *Então, para alguma função mensurável* $g : \mathbb{R} \to \mathbb{R}$,

$$E\{g(U)|\mathbf{X} = \mathbf{x}\} = 2\,\boldsymbol{\gamma}(\mathbf{x}) E\{g(U_x)\boldsymbol{\Phi}(U_x^{1/2}\,\boldsymbol{\lambda}^T\boldsymbol{\Sigma}^{-1/2}(\mathbf{x} - \boldsymbol{\mu}))\},$$

onde $\boldsymbol{\gamma}(\mathbf{x}) = NI_p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, H)/NAI_p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, H)$.

Este resultado nos permite calcular a integral $E\{g(U)|\mathbf{X} = \mathbf{x}\}$ de uma forma muita mais simples, pois integramos com relação a distribuição de U_x. Para mais detalhes sobre estas proposições ver Cabral *et al.* [5] e suas referências para a demonstração das mesmas.

A família NAI inclui distribuições assimétricas tais como a normal assimétrica, a *t* assimétrica, a slash assimétrica e a normal contaminada, e suas simétricas equivalentes. As distribuições normal assimétrica e t assimétrica baseadas na definição acima, são apresentadas a seguir.

3.1.1 Distribuição Normal Assimétrica e t Assimétrica Multivariadas

Fazendo $U = 1$ e resolvendo (**??**) obtemos a densidade

$$NA_p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}) = 2\phi_p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\Phi(\,\boldsymbol{\lambda}^T\boldsymbol{\Sigma}^{-1/2}(\mathbf{x} - \boldsymbol{\mu})), \tag{3.5}$$

que por definição é a forma da distribuição normal assimétrica. Como mencionado, com $\boldsymbol{\lambda} = 0$ nós obtemos a distribuição $N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Tomando U com distribuição $Gama(\nu/2, \nu/2)$, $\nu > 0$ e resolvendo (**??**) obtemos a distribuição t assimétrica, cuja densidade é da forma

$$tA_p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\lambda}, \nu) = 2t_p(\mathbf{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \nu) T\left(\sqrt{\frac{\nu + p}{\nu + d_\Sigma(\mathbf{x}, \boldsymbol{\mu})}}\,\boldsymbol{\lambda}^T\boldsymbol{\Sigma}^{-1/2}(\mathbf{x} - \boldsymbol{\mu})|\nu + p\right), \tag{3.6}$$

onde $t_p(.|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \nu)$ denota a distribuição t multivariada, de dimensão p, com vetor de

médias $\boldsymbol{\mu}$, matriz de escala $\boldsymbol{\Sigma}$ e ν graus de liberdade, e $T(.|\nu+p)$ denota a função de distribuição da t univariada padronizada, com $\nu+p$ graus de liberdade. O termo $d_{\Sigma}(\mathbf{x},\boldsymbol{\mu})=(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})$ é o quadrado da distância de Mahalanobis entre $\mathbf{x}$ e $\boldsymbol{\mu}$ com respeito a $\boldsymbol{\Sigma}$. Note que, com $\boldsymbol{\lambda}=0$ nós obtemos a distribuição t multivariada. E também, quando $\nu\uparrow\infty$ nós obtemos a distribuição normal assimétrica como limite.

3.1.2 Estimação de parâmetros via algoritmo EM

Para estimação dos parâmetros das distribuições na família NAI empregamos o método da máxima verossímilhança e solucionamos as equações deste método implementando um algoritmo do tipo EM.

Para o desenvolvimento do algoritmo, seguindo Cabral *et al.* [5], sejam $(\mathbf{x}_i^T,\mathbf{t}_i,\mathbf{u}_i)^T$ os dados completos, uma amostra aleatória de $(\mathbf{X},\mathbf{T},\mathbf{U})$, e $\mathbf{b}$ os elementos da matriz triangular superior $\boldsymbol{\Sigma}$. Sendo $\mathbf{x}=(\mathbf{x}_1^T,...,\mathbf{x}_n^T)^T$, $\mathbf{t}=(t_1,...,t_n)^T$ e $\mathbf{u}=(u_1,...u_n)^T$, a função log-verossimilhança para os dados completos de $\boldsymbol{\theta}=(\boldsymbol{\mu},\mathbf{b},\ \boldsymbol{\lambda},\boldsymbol{\nu})^T$ é dada por

$$l_c(\boldsymbol{\theta}|\mathbf{x},\mathbf{t},\mathbf{u})=K-\frac{n}{2}\log|\boldsymbol{\Gamma}|+\sum_{i=1}^{n}\log(h(u_i|\boldsymbol{\nu}))-\frac{1}{2}\sum_{i=1}^{n}u_i(\mathbf{x}_i-\boldsymbol{\mu}-\boldsymbol{\Delta}t_i)^T\boldsymbol{\Gamma}^{-1}(\mathbf{x}_i-\boldsymbol{\mu}-\boldsymbol{\Delta}t_i), \quad (3.7)$$

onde K é uma constante independente do vetor de parâmetros $\boldsymbol{\theta}$. Na etapa E do algoritmo incluímos esses dados não observados ao problema e obtemos a esperança de (**??**), dado o vetor de dados observados,

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(k)})=E\{l_c(\boldsymbol{\theta}|\mathbf{x},\mathbf{t},\mathbf{u})|\mathbf{x}\}, \quad (3.8)$$

Para determinarmos a expressão da função Q de (**??**) são necessárias as esperanças condicionais:

$$\begin{aligned}\beta(\boldsymbol{\theta}^{(k)},\mathbf{x}_i) &= E_{\boldsymbol{\theta}^{(k)}}\{U_i|\mathbf{x}_i\},\\ \xi(\boldsymbol{\theta}^{(k)},\mathbf{x}_i) &= E_{\boldsymbol{\theta}^{(k)}}\{U_iT_i|\mathbf{x}_i\}\end{aligned}$$

e

$$\omega(\boldsymbol{\theta}^{(k)},\mathbf{x}_i) = E_{\boldsymbol{\theta}^{(k)}}\{U_iT_i^2|\mathbf{x}_i\}. \quad (3.9)$$

A primeira expressão em (**??**) pode ser obtida empregando a proposição (**??**), fazendo

$g(u) = u$. Para obter $\xi(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)$ e $\omega(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)$, note que, pela mesma proposição,

$$T_i|\mathbf{x}_i, u_i \sim NT(m(\boldsymbol{\theta}, \mathbf{x}_i), u_i^{-1}M^2(\boldsymbol{\theta}), (0, \infty)),$$

com $M^2(\boldsymbol{\theta}) = (1 + \boldsymbol{\Delta}^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\Delta})^{-1}$ e $m(\boldsymbol{\theta}, \mathbf{x}_i) = M^2(\boldsymbol{\theta})\boldsymbol{\Delta}^T\boldsymbol{\Gamma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu})$.

Baseado nestes resultados, o desenvolvimento para obter as esperanças condicionais em (**??**) está descrito em Basso [4] e foram obtidas as seguintes expressões:

$$\begin{aligned} \xi(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i) &= \beta(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)m(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i) + M(\boldsymbol{\theta}^{(k)})\tau(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i) \quad \text{e} \\ \omega(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i) &= \beta(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)(m(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i))^2 + (M(\boldsymbol{\theta}^{(k)}))^2 + M(\boldsymbol{\theta}^{(k)})m(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)\tau(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i), \end{aligned} \tag{3.10}$$

tal que

$$\tau(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i) = E_{\boldsymbol{\theta}^{(k)}}\{U_i^{1/2}W_\Phi(U_i^{1/2}A(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i))|\mathbf{x}_i\}. \tag{3.11}$$

onde $A(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)) = m(\boldsymbol{\theta}, \mathbf{x}_i)/M(\boldsymbol{\theta})$ e $W_\Phi(\cdot) = \phi(\cdot)/\Phi(\cdot)$. A esperança (**??**) pode ser obtida usando a proposição (**??**), fazendo $g(u) = u^{1/2}W_\Phi(u_i^{1/2}A(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i))$.

Portanto, o valor esperado condicional da log-verossimilhança para os dados completos é dado por

$$\begin{aligned} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(k)}) &= E_{\boldsymbol{\theta}^{(k)}}\{l_c(\boldsymbol{\theta}|\mathbf{x}, \mathbf{t}, \mathbf{u})|\mathbf{x}\} \\ &= K - \frac{n}{2}\log|\boldsymbol{\Gamma}| - \frac{1}{2}\sum_{i=1}^{n}\{(\mathbf{x}_i - \boldsymbol{\mu})^T\boldsymbol{\Gamma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu})\hat{\beta}(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i) - 2\boldsymbol{\Delta}^T\boldsymbol{\Gamma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu})\hat{\xi}(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i) \\ &\quad + \boldsymbol{\Delta}^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\Delta}^T\hat{\omega}(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)\} + \sum_{i=1}^{n}E[\log(h(u_i|\boldsymbol{\nu}))], \end{aligned} \tag{3.12}$$

No passo M é realizada a maximização da log-verossimilhança completa em relação aos parâmetros do modelo, substituindo os dados latentes por seus valores esperados condicionais obtidos na etapa E. Nesta etapa, é usada uma extensão do algoritmo EM, chamada algoritmo ECME, que substitui o passo M por uma sequência de maximizações condicionais (ver Liu & Rubin [17]). Os passos do algoritmo ficam então definidos como:

Passo E: Dado $\boldsymbol{\theta} = \boldsymbol{\theta}^{(k)}$, calcule $\beta_i^{(k)} = \beta(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)$, $\xi_i^{(k)} = \xi(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)$ e $\omega_i^{(k)} = \omega(\boldsymbol{\theta}^{(k)}, \mathbf{x}_i)$, para $i = 1, ..., n$.

Passo M: 1.Atualize $\boldsymbol{\mu}^{(k)}$, $\boldsymbol{\Gamma}^{(k)}$ e $\boldsymbol{\Delta}^{(k)}$ usando as seguintes expressões fechadas

$$\begin{aligned}
\boldsymbol{\mu}^{(k+1)} &= \sum_{i=1}^{n}(\beta_i^{(k)}\mathbf{x}_i - \xi_i^{(k)}\boldsymbol{\Delta}^{(k)})/(\sum_{i=1}^{n}\beta_i^{(k)}),\\
\boldsymbol{\Delta}^{(k+1)} &= \sum_{i=1}^{n}\xi_i^{(k)}(\mathbf{x}_i - \boldsymbol{\mu}_i^{(k+1)})/(\sum_{i=1}^{n}\omega_i^{(k)}),\\
\boldsymbol{\Gamma}^{(k+1)} &= \frac{1}{n}\sum_{i=1}^{n}\{\beta_i^{(k)}(\mathbf{x}_i - \boldsymbol{\mu}^{(k+1)})(\mathbf{x}_i - \boldsymbol{\mu}^{(k+1)})^T - \xi_i^{(k)}[\boldsymbol{\Delta}^{(k+1)}(\mathbf{x}_i - \boldsymbol{\mu}^{(k+1)})^T\\
&\quad + (\mathbf{x}_i - \boldsymbol{\mu}^{(k+1)})(\boldsymbol{\Delta}^{(k+1)})^T] + \omega_i^{(k)}\boldsymbol{\Delta}^{(k+1)}(\boldsymbol{\Delta}^{(k+1)})^T\}.
\end{aligned}$$

2. Atualize $\boldsymbol{\nu}^{(k)}$ maximizando a função log-verossimilhança marginal no passo k,

$$\boldsymbol{\nu}^{(k+1)} = \arg\max_{\boldsymbol{\nu}} \sum_{i=1}^{n} \log(NAI(\mathbf{x}_i|\boldsymbol{\mu}^{(k+1)}, \boldsymbol{\Sigma}^{(k+1)}, \boldsymbol{\lambda}^{(k+1)}, \boldsymbol{\nu})). \tag{3.13}$$

No segundo momento do passo M, as atualizações de $\boldsymbol{\Sigma}^{(k+1)}$ e $\boldsymbol{\lambda}^{(k+1)}$ são obtidas pelas expressões em (**??**). Esses passos são repetidos alternadamente até satisfazer a regra de convergência empregada.

Na literatura temos algumas propostas para a escolha dos valores iniciais do algoritmo. Uma delas, por exemplo, é selecionar aleatoriamente um determinado número de observações dentro da amostra $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$, para usá-las como $\boldsymbol{\mu}^{(0)}$ e para $\boldsymbol{\Sigma}^{(0)}$ empregar a matriz identidade. Para uma discussão mais detalhada ver McLachlan & Peel [19]. Esta questão será abordada na Seção **??**, visando os interesses deste trabalho.

3.2 Misturas Finitas de NAI Multivariadas

Definiremos, segundo Cabral *et al.* [5], o modelo de mistura finita de NAI (MF-NAI) fazendo uma extensão do que foi apresentado nas seções anteriores. Seja $\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_n$ uma amostra aleatória de uma mistura finita de densidades na família NAI com g componentes, sua densidade é dada por

$$f(\mathbf{x}|\boldsymbol{\Theta}) = \sum_{j=1}^{g} p_i NAI_p(\mathbf{x}|\boldsymbol{\theta}_j), \quad p_j \geq 0, \quad \sum_{j=1}^{g} p_j = 1, \tag{3.14}$$

onde $\boldsymbol{\Theta} = ((\theta_1^T, p_1), ..., (\boldsymbol{\theta}_g^T, p_g))^T$, com $\boldsymbol{\theta}_j = (\boldsymbol{\mu}_j^T, \boldsymbol{\alpha}_j^T, \lambda_j^T, \nu_j)^T$ no caso da t assimétrica, e $\boldsymbol{\theta}_j = (\boldsymbol{\mu}_j^T, \boldsymbol{\alpha}_j^T, \boldsymbol{\lambda}_j^T)^T$ no caso da normal assimétrica, $j = 1, ..., g$, o vetor de parâmetros específicos de cada componente, e $p_j's$ são os pesos da mistura.

Na Figura **??** apresentamos um exemplo de uma mistura finita de normais assimétricas univariadas, com $g = 3$. Observe sua assimetria e multimodalidade.

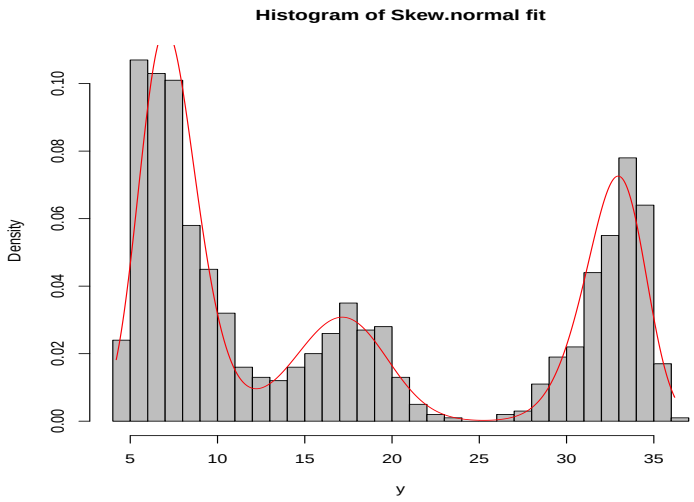


Figura 3.3: Gráfico de uma MF de densidades de NA univariada.

Na Figura **??** apresentamos um exemplo de uma mistura finita de t assimétricas bivariadas, com $g = 2$. Aqui também podemos observar assimetria e multimodalidade.

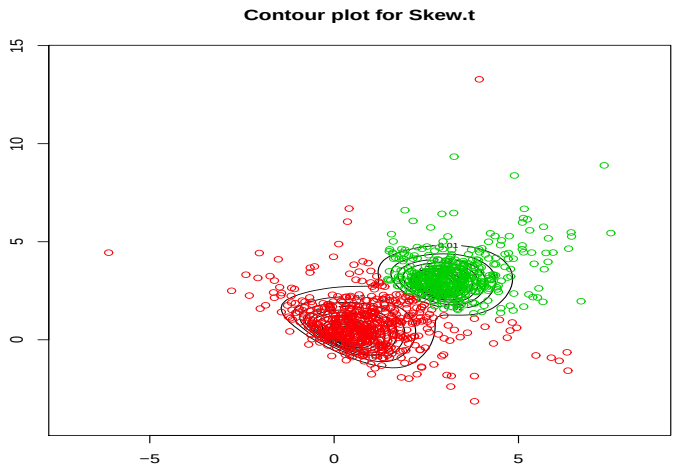


Figura 3.4: Dados ajustados por uma MF de tA bivariada.

Como visto na Seção 1.3, para estimação dos parâmetros no contexto de mistura de densidades via um algoritmo do tipo EM , introduzimos o vetor de dados latentes, não

observados, $\mathbf{Z}_i = (Z_{i1}, ..., Z_{ig})$ cuja finalidade é associar a i-ésima observação da amostra a uma das g componentes da mistura considerada. Temos que a distribuição de $\mathbf{X}_i|Z_{ij}$ é $NAI_p(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \boldsymbol{\lambda}_j, \boldsymbol{\nu})$ e $\mathbf{Z} \sim Multi_g(1; p_1, ..., p_g)$. Usando a proposição (**??**), podemos representar hierarquicamente o modelo MF-NAI, tendo como objetivo facilitar a estimação dos parâmetros, uma vez que trabalhamos com distribuições conhecidas. Assim, temos

$$\begin{aligned} \mathbf{X}_i|U_i = u_i, T_i = t_i, Z_{ij} = 1 &\sim N_p(\boldsymbol{\mu}_j + \boldsymbol{\Delta}_j t_i, u_i^{-1}\boldsymbol{\Gamma}_j), \\ T_i|U_i = u_i, Z_{ij} = 1 &\sim HN(0, u_i^{-1}), \\ U_i|Z_{ij} = 1 &\sim H(u_i|\boldsymbol{\nu}), \\ Z_i &\sim Multi_g(1; p_1....p_g), \end{aligned} \tag{3.15}$$

onde $i = 1, ..., n$ e $j = 1, ..., g$, $HN(0, u_i^{-1})$ denota a distribuição *half-normal* com média zero e variancia u_i^{-1} (antes de truncar em $(0, \infty)$) e

$$\boldsymbol{\Delta}_j = \boldsymbol{\Sigma}_j^{1/2}\boldsymbol{\delta}_j, \ \boldsymbol{\Gamma}_j = \boldsymbol{\Sigma}_j - \boldsymbol{\Delta}_j\boldsymbol{\Delta}_j^T.$$

Em particular, neste trabalho, serão consideradas misturas finitas de densidades normais assimétricas (MFNA) e misturas finitas de densidades t assimétricas (MFtA), dadas, respectivamente, por

$$f(\mathbf{x}|\Theta_j) = \sum_{j=1}^{g} p_j NA_p(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \boldsymbol{\lambda}_j) \tag{3.16}$$

e

$$f(\mathbf{x}|\Theta_j) = \sum_{j=1}^{g} p_j tA_p(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \boldsymbol{\lambda}_j, \nu_j). \tag{3.17}$$

A estimação dos parâmetros desses modelos é pelo método da máxima verossimilhança, por meio de um algoritmo do tipo EM, semelhante ao apresentado na seção **??**. A seguir, são definidas então as equações e os passos do algoritmo no contexto de misturas finitas de NAI.

3.3 Estimação dos parâmetros das MF-NAI

Sejam $(\mathbf{x}_i^T, \mathbf{t}_i, \mathbf{u}_i)^T$ os dados completos, uma amostra aleatória de $(\mathbf{X}, T, U)$, e $\mathbf{b}$ os elementos da matriz triangular superior $\boldsymbol{\Sigma}$. Sendo $\mathbf{x} = (\mathbf{x}_1^T, ..., \mathbf{x}_n^T)^T$, $\mathbf{t} = (t_1, ..., t_n)^T$ e

$\mathbf{u} = (u_1, \ldots u_n)^T$, a função log-verossimilhança para os dados completos de $\boldsymbol{\theta} = (\boldsymbol{\mu}, \mathbf{b}, \boldsymbol{\lambda}, \boldsymbol{\nu})^T$ é dada por

$$\begin{aligned} l_c(\boldsymbol{\theta}|\mathbf{x},\mathbf{t},\mathbf{u},\mathbf{z}) &= K + \sum_{i=1}^{n}\sum_{j=1}^{g} Z_{ij}\Big[\log(p_j) - \frac{1}{2}\log|\boldsymbol{\Gamma}_j| \\ &\quad -\frac{u_i}{2}(\mathbf{x}_i - \boldsymbol{\mu}_j - \boldsymbol{\Delta}_j\mathbf{t}_i)^T\boldsymbol{\Gamma}_j^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_j - \boldsymbol{\Delta}_j\mathbf{t}_i) \\ &\quad +\log(h(u_i|\boldsymbol{\nu}))\Big] \end{aligned} \tag{3.18}$$

onde K é uma constante independente de $\boldsymbol{\theta}$ e $\mathbf{z} = (\mathbf{z}_1^T, \ldots, \mathbf{z}_n^T)^T$.

No passo E, as esperanças condicionais envolvidas para encontrar a solução de $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) = E_{\boldsymbol{\theta}^{(k)}}\{l_c(\boldsymbol{\Theta}|\mathbf{x},\mathbf{t},\mathbf{u},\mathbf{z})|\mathbf{x}\}$ são

$$\tau_{ij}^{(k)} = E_{\boldsymbol{\Theta}^{(k)}}\{Z_{ij}|\mathbf{x}_i\}, \tag{3.19}$$

$$\beta_{ij}^{(k)} = E_{\boldsymbol{\Theta}^{(k)}}\{Z_{ij}U_i|\mathbf{x}_i\}, \tag{3.20}$$

$$\xi_{ij}^{(k)} = E_{\boldsymbol{\Theta}^{(k)}}\{Z_{ij}U_iT_i|\mathbf{x}_i\} \quad \text{e} \tag{3.21}$$

$$\omega_{ij}^{(k)} = E_{\boldsymbol{\Theta}^{(k)}}\{Z_{ij}U_iT_i^2|\mathbf{x}_i\} \tag{3.22}$$

Da Seção 1.3 temos que

$$\tau_{ij}^{(k)} = \frac{p_j^{(k)}NAI(\mathbf{x}_i;\boldsymbol{\theta}_j^{(k)})}{\sum_{t=1}^{g} p_t^{(k)}NAI(\mathbf{x}_i;\boldsymbol{\theta}_t^{(k)})}. \tag{3.23}$$

Usando propriedades de esperança condicional e (**??**), segundo o desenvolvimento feito em Basso [4], temos que

$$\begin{aligned} \beta_{ij}^{(k)} &= \tau_{ij}^{(k)}\beta(\boldsymbol{\theta}^{(k)},\mathbf{x}_i), \\ \xi_{ij}^{(k)} &= \tau_{ij}^{(k)}\xi(\boldsymbol{\theta}^{(k)},\mathbf{x}_i) \quad \text{e} \\ \omega_{ij}^{(k)} &= \tau_{ij}^{(k)}\omega(\boldsymbol{\theta}^{(k)},\mathbf{x}_i). \end{aligned} \tag{3.24}$$

Portanto, a verossimilhança em (**??**) é dada por

$$\begin{aligned} l_c(\boldsymbol{\theta}|\mathbf{x},\mathbf{t},\mathbf{u},\mathbf{z}) &= K+\sum_{i=1}^{n}\sum_{j=1}^{g}\hat{\tau}_{ij}\Big[\log(p_j)-\frac{1}{2}\log|\boldsymbol{\Gamma}_j| \\ &\quad -\frac{1}{2}\hat{\beta}_{ij}(\mathbf{x}_i-\boldsymbol{\mu}_j)^T\boldsymbol{\Gamma}_j^{-1}(\mathbf{x}_i-\boldsymbol{\mu}_j) \\ &\quad +\hat{\xi}_{ij}(\mathbf{x}_i-\boldsymbol{\mu}_j)^T\boldsymbol{\Gamma}_j^{-1}\boldsymbol{\Delta}_j+\frac{1}{2}\hat{\omega}_{ij}\boldsymbol{\Delta}_j^T\boldsymbol{\Gamma}_j^{-1}\boldsymbol{\Delta}_j+\log(h(u_i|\boldsymbol{\nu}))\Big] \end{aligned} \tag{3.25}$$

Como foi apresentado na Subseção **??** para o caso de uma componente, estendemos o algoritmo do tipo EM para um algoritmo do tipo ECME. Os passos do algoritmo ficam então definidos como:

Passo E: Dado $\boldsymbol{\Theta}=\boldsymbol{\Theta}^{(k)}$, calcule $\tau_{ij}^{(k)}$, $\beta_{ij}^{(k)}$, $\xi_{ij}^{(k)}$ e $\omega_{ij}^{(k)}$ para $i=1,...,n$ e $j=1,...,g$.

Passo M: 1.Para $j=1,...,g$, atualize $p_j^{(k)}$, $\boldsymbol{\mu}_j^{(k)}$, $\boldsymbol{\Gamma}_j^{(k)}$ e $\boldsymbol{\Delta}_j^{(k)}$ usando as seguintes expressões fechadas:

$$\begin{aligned} p_j^{(k+1)} &= n^{-1}\sum_{i=1}^{n}\tau_{ij}^{(k)} \\ \boldsymbol{\mu}_j^{(k+1)} &= \sum_{i=1}^{n}(\beta_{ij}^{(k)}\mathbf{x}_i-\xi_{ij}^{(k)}\boldsymbol{\Delta}_j^{(k)})/(\sum_{i=1}^{n}\beta_{ij}^{(k)}), \\ \boldsymbol{\Delta}_j^{(k+1)} &= \sum_{i=1}^{n}\xi_{ij}^{(k)}(\_i-\boldsymbol{\mu}_j^{(k+1)})/(\sum_{i=1}^{n}\omega_{ij}^{(k)}), \\ \boldsymbol{\Gamma}_j^{(k+1)} &= \Big(\sum_{i=1}^{n}\tau_{ij}^{(k)}\Big)^{-1}\sum_{i=1}^{n}(\beta_{ij}^{(k)}(\mathbf{x}_i-\boldsymbol{\mu}_j^{(k+1)})(\mathbf{x}_i-\boldsymbol{\mu}_j^{(k+1)})^T \\ &\quad -[(\mathbf{x}_i-\boldsymbol{\mu}_j^{(k+1)})(\boldsymbol{\Delta}_j^{(k+1)})^T+(\boldsymbol{\Delta}_j^{(k+1)})(\mathbf{x}_i-\boldsymbol{\mu}_j^{(k+1)})^T]\xi_{ij}^{(k)} \\ &\quad +\boldsymbol{\Delta}_j^{(k+1)}(\boldsymbol{\Delta}_j^{(k+1)})^T\omega_{ij}^{(k)}). \end{aligned}$$

2. Atualize $\boldsymbol{\nu}^{(k)}$ maximizando a função log-verossimilhança marginal no passo k,

$$\boldsymbol{\nu}^{(k+1)}=\arg\max_{\boldsymbol{\nu}}\sum_{i=1}^{n}\log(\sum_{j=1}^{g}p_j NAI(\mathbf{x}_i|\boldsymbol{\mu}^{(k+1)},\boldsymbol{\Sigma}^{(k+1)},\boldsymbol{\lambda}^{(k+1)},\boldsymbol{\nu})). \tag{3.26}$$

As iterações são repetidas até que a regra de convergência empregada seja satisfeita,

que neste trabalho consideramos como

$$\left\| \frac{l(\boldsymbol{\theta}^{(k+1)})}{l(\boldsymbol{\theta}^{(k)}) - 1} \right\| < 10^{-4}.$$

Note que, é necessário o valor das esperanças condicionais $\tau(\boldsymbol{\theta}, \mathbf{x})$, $\beta(\boldsymbol{\theta}, \mathbf{x})$, $\xi(\boldsymbol{\theta}, \mathbf{x})$ e $\omega(\boldsymbol{\theta}, \mathbf{x})$. Mas por (**??**), precisamos apenas encontrar $\beta(\boldsymbol{\theta}, \mathbf{x})$ e $\tau(\boldsymbol{\theta}, \mathbf{x})$, uma vez que $\xi(\boldsymbol{\theta}, \mathbf{x})$ e $\omega(\boldsymbol{\theta}, \mathbf{x})$ são facilmente obtidas dado o valor de $\beta(\boldsymbol{\theta}, \mathbf{x})$ e $\tau(\boldsymbol{\theta}, \mathbf{x})$. Definiremos estas esperanças condicionais a seguir, onde especificaremos as distribuições de interesse para este trabalho pertencentes à família NAI e as suas respectivas expressões.

3.3.1 Passo E para Misturas de Normais Assimétricas

No caso da distribuição normal assimétrica,

$$\begin{aligned} \beta(\boldsymbol{\theta}, \mathbf{x}) &= 1, \\ \tau(\boldsymbol{\theta}, \mathbf{x}) &= W_{\Phi}(A(\boldsymbol{\theta}, \mathbf{x})), \end{aligned}$$

onde $W_{\Phi}(\cdot) = \phi(\cdot)/\Phi(\cdot)$ e $A(\boldsymbol{\theta}, \mathbf{x}_i) = m(\boldsymbol{\theta}, \mathbf{x}_i)/M(\boldsymbol{\theta})$.

3.3.2 Passo E para Misturas de t Assimétricas

No caso da distribuição t assimétrica, usando a notação da Proposição 3.2 e

$$\begin{aligned} \mathbf{X}_0 | U = u &\sim N_p(\boldsymbol{\mu}, u^{-1}\Sigma), \\ U &\sim Gama(\nu/2, \nu/2), \end{aligned}$$

o que resulta em $U_i \overset{d}{=} (U|\mathbf{X}_0 = \mathbf{x}) \sim Gama((\nu + p)/2)$, após alguns passos algébricos, temos

$$\begin{aligned} \beta(\boldsymbol{\theta}, \mathbf{x}) &= \frac{2\Gamma(\frac{\nu+p+2}{2})(\nu + d_{\Sigma}(\mathbf{x}, \boldsymbol{\mu}))^{-1}}{\Gamma(\frac{\nu+p}{2})} \frac{T\left(\sqrt{\frac{\nu+p+2}{\nu+d_{\Sigma}(\mathbf{x}, \boldsymbol{\mu})}} A(\boldsymbol{\theta}, \mathbf{x}) | \nu + p + 2\right)}{T\left(\sqrt{\frac{\nu+p}{\nu+d_{\Sigma}(\mathbf{x}, \boldsymbol{\mu})}} A(\boldsymbol{\theta}, \mathbf{x}) | \nu + p\right)} \\ \tau(\boldsymbol{\theta}, \mathbf{x}) &= \frac{1}{T\left(\sqrt{\frac{\nu+p}{\nu+d_{\Sigma}(\mathbf{x}, \boldsymbol{\mu})}} A(\boldsymbol{\theta}, \mathbf{x} | \nu + p)\right)} \frac{\Gamma\left(\frac{\nu+p+1}{2}\right)}{\pi^{1/2}\Gamma(\frac{\nu+p}{2})} \frac{(\nu + d_{\Sigma}(\mathbf{x}, \boldsymbol{\mu}))^{(\nu+p)/2}}{(\nu + d_{\Sigma}(\mathbf{x}, \boldsymbol{\mu}) + A(\theta, \mathbf{x})^2)^{(\nu+p+1)/2}}, \end{aligned}$$

onde $T(.|\nu+p)$ denota a função de distribuição da t-Student padronizada univariada , com $\nu+p$ graus de liberdade, $A(\boldsymbol{\theta},\mathbf{x}_i) = m(\boldsymbol{\theta},\mathbf{x}_i)/M(\boldsymbol{\theta})$ e $d_\Sigma(\mathbf{x},\boldsymbol{\mu}) = (\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})$ é a raiz da distância de Mahalanobis entre $\mathbf{x}$ e $\boldsymbol{\mu}$ com respeito a $\boldsymbol{\Sigma}$.

Como citado anteriormente, para a implementação do algoritmo, é necessário um valor inicial para $\boldsymbol{\Theta}$, $\boldsymbol{\Theta}^{(0)}$. Neste trabalho os valores iniciais foram obtidos a partir de um agrupamento inicial das observações, realizado pelo *k-means* (ver Hartigan & Wong [9]). Definidos os grupos então, tomamos como valores iniciais dos parâmetros nas componentes das misturas as estimativas de máxima verossimilhança obtidas das obsservações em cada grupo.

Este método para obtenção de valores iniciais do algoritmo EM é bastante conhecido, suas propriedades e eficiência são relatadas na literatura (Ripley [24]) e, como exemplo, podemos citar Wang & Hu [27] que realiza o mesmo procedimento proposto neste trabalho. Espera-se, então, que os valores iniciais fornecidos contribuam para acelerar a convergência do algoritmo.

Capítulo 4

Análise Discriminante

Neste capítulo discutimos o problema de classificação de objetos empregando a modelagem estatística. Definimos Análise Discriminante, o classificador de Bayes e de máxima verossimilhança e, também, o critério estatístico usual para avaliar a performance das regras de classificação.

4.1 Análise Discriminante

Os problemas em Análise Discriminante (AD), como mencionado no Capítulo 1, consistem em alocar *objetos* a *classes* previamente definidas. O termo *objeto* é empregado para denotar o que temos interesse em alocar nas possíveis categorias, as *classes*, definidas para o problema em questão. O objeto é descrito por uma coleção de variáveis que, consideradas conjuntamente, recebe a denominação de *vetor de características* e as variáveis que o compõem são denominadas *variáveis preditoras*. A classificação de um objeto em uma das classes é efetuada com base no valor observado do seu vetor de características. Portanto, é necessário desenvolver um procedimento que, baseado na informação contida no vetor de características de um objeto, indique em qual classe alocar este objeto. Tal procedimento é denominado de *classificador*, às vezes também denominado de *função discriminante*.

Considere um problema com M classes, denotadas por $C_1, C_2, C_3, ..., C_M$. Denotando por $\mathbf{X}^T = (X_1, ..., X_p)$ o vetor de características e por $\mathbf{x}$ seu valor observado em um dado objeto, definimos:

Definição 4.1. *Um classificador (ou função discriminante) é qualquer função $r(\cdot)$ para*

a qual temos $r : \mathbb{R}^p \longrightarrow \{1,2,3,...,M\}$.

Pelos objetivos da AD, entre outras etapas, devemos obter um classificador $r(\cdot)$ e avaliar seu desempenho com relação a alocar de forma correta os objetos as suas respectivas classes. Um bom classificador produz poucos erros de classificação, isto é, do ponto de vista estatístico, sua probabilidade de erro de classificação deve ser pequena.

Em AD dispomos de amostras aleatórias que são identificadas com relação à classe de onde foram observadas. Este conjunto de observações "rotuladas" é denominado de *conjunto de treinamento*. Com $Y \in \{1,2,3,...,M\}$, os objetos são descritos por um par de variáveis $(\mathbf{X},Y)$ e o conjunto de treinamento é da forma

$$\mathscr{T}_{(n)} = \big\{(\mathbf{X}_1,Y_1),(\mathbf{X}_2,Y_2),(\mathbf{X}_3,Y_3),...,(\mathbf{X}_n,Y_n)\big\},$$

onde temos n_j observações para os quais $Y = j$, isto é, o número de observações da classe C_j, $j = 1,...,M$. Temos portanto, $\sum_{j=1}^{M} n_j = n$.

Uma abordagem estatística consiste em modelar o vetor de características por uma distribuição de probabilidade, condicionada a cada uma das classes. Se considerarmos modelos paramétricos para estas distribuições, as amostras disponíveis são empregadas para estimar os parâmetros envolvidos. Como será visto, estas distribuições estimadas são empregadas para construirmos o classificador, com finalidade de classificar futuras observações cujas classes sejam desconhecidas.

Pode ser que observações de uma determinada classe tenham uma maior probabilidade de ocorrência do que uma outra, às vezes denominada de classe com *maior prevalência*. Se dispormos de informação sobre as probalidades de ocorrências das classes, é razoável considerarmos na construção do classificador essas probabilidades. Esta é a idéia subjacente ao desenvolvimento do *classificador de Bayes*. Em muitas aplicações, no entanto, não dispomos do conhecimento sobre o valor destas probabilidades e, nestes casos, podemos considerar um classificador baseado simplesmente em um procedimento de *verossimilhança*. Esta situação se verifica, por exemplo, em um dos problemas com dados reais analisado neste trabalho, descrito na Seção **??**, que consiste em classificar *pixels* de uma imagem digital de material de exame de baciloscopia como sendo de um bacilo da tuberculose ou não.

Outro aspecto é a perda resultante do processo de classificação. Suponha que classificar um objeto que pertence a uma classe, denotada por C_1, como pertencente à outra classe, denotada por C_2, representa um erro mais grave do que classificando o objeto de

C_2 como pertencente à C_1. Por exemplo, deixar de diagnosticar uma doença grave tem um custo maior do que concluir que a doença está presente quando, de fato, não está. Uma regra de classificação ideal deve, sempre que possível, considerar as perdas associadas a uma má classificação. Em muitos problemas reais, no entanto, também não é possível estabelecer corretamente os custos associados.

Neste trabalho, o classificador utilizado nas simulações foi o *classificador de Bayes* e para os dados reais empregamos o *classificador de Máxima Verossimilhança*, uma vez que desconhecemos as probabilidades *a priori* das classes. Esses classificadores serão discutidos na próxima seção.

4.2 Classificador de Bayes e de Máxima Verossimilhança

O desenvolvimento do classificador de Bayes é baseado nos procedimento da Teoria da Decisão. Sejam $C_1, C_2, C_3, ..., C_M$ as M classes definidas num problema e seja $\mathbf{X}$ o vetor de características que assume valores em $\mathbb{R}^p$ que, como já mencionado, será modelado como um vetor aleatório. Seja $Y \in \{1, 2, 3, ..., M\}$ uma variável indicadora das classes para os objetos. Em cada classe C_j, $j = 1, 2, 3, .., M$, $\mathbf{X}$ é modelado por uma distribuição $f_j(\cdot|Y = j)$ adequada às variáveis preditoras. Esssas distribuições são denominadas *densidades condicionais das classes*. Seja $P(C_j) = P(Y = j)$ a probabilidade de um objeto provir da classe C_j, que é denominada *probabilidade a priori da classe j*, para $j = 1, 2, .., M$. Desta forma, o vetor de características para objetos é modelado como um par aleatório $(\mathbf{X}, Y)$.

Com os termos definidos acima, dado $\mathbf{X} = \mathbf{x}$, empregamos o Teorema de Bayes para determinar a *probabilidade a posteriori* da classe C_j, que é dada por

$$P(C_j|\mathbf{x}) = P(Y = j|\mathbf{X} = \mathbf{x}) = \frac{f(\mathbf{x}|Y = j)P(Y = j)}{\sum_{l=1}^{M} f(\mathbf{x}|Y = l)P(Y = l)}, \quad j = 1, ..., M. \tag{4.1}$$

A idéia é empregar (**??**) para construir o classificador.

Pela Definição (**??**), dado um classificador r e um objeto para o qual observamos $\mathbf{X} = \mathbf{x}$, $r(\mathbf{x}) = j$ significa que o objeto é alocado para a classe C_j. Note que, existe uma infinidade de classificadores para um problema, fazendo-se necessário um critério para avaliá-los.

Segundo a Teoria da Decisão, uma maneira usual de formalizar um critério para avaliar

classificadores é estabelecer uma *função de perda*. Para uma função de perda $\lambda(\cdot,\cdot)$, seja $\lambda(i,j)$ a perda decorrente de alocar um objeto da classe C_i para a classe C_j, isto é, a perda decorrente do processo de alocação. Temos que $\lambda(i,i)=0$, para $i=1,2,3,...,M$. Em muitas aplicações, onde não é possível ou não há necessidade de especifiar as perdas $\lambda(i,j)$, os custos de má classificação podem ser considerados iguais, sendo comum adotar

$$\lambda(i,j)=\begin{cases} 0 & \text{se } i=j \\ 1 & \text{se } i\neq j \end{cases} \tag{4.2}$$

A função definida em (**??**) é denominada *função de perda* $0-1$.

Note que, considerando a modelagem adotada, temos que $\lambda(i,j)=\lambda(i,r(\mathbf{X})=j)$ para $j=1,2,..,M$, portanto, trata-se de uma variável aleatória e devemos considerá-la em termos de seu valor esperado como um critério para desenvolver um classificador. Para esse fim, considerando uma especificada função de perda, empregamos a definição de *função de risco* e do *risco total*.

Definição 4.2. *Para um classificador r, com uma função de perda $\lambda(\cdot,\cdot)$, definimos:*
(a) A Função de Risco *é a perda esperada como função de uma classe C_i, ou seja*

$$\begin{aligned} R(r,i) &= E\{\lambda(i,r(\mathbf{X}))|Y=i\} \\ &= \sum_{i\neq j=1}^{M}\lambda(i,j)P(r(\mathbf{X})=j|Y=i). \end{aligned} \tag{4.3}$$

(b) O Risco total*, ou* Risco de *r, é a perda total esperada como função das variáveis aleatórias* $\mathbf{X}$ *e* Y*, ou seja,*

$$\begin{aligned} R(r) &= E\{R(r,Y)\} \\ &= \sum_{i=1}^{M}R(r,i)P(Y=i) \\ &= \sum_{i=1}^{M}\sum_{i\neq j=1}^{M}\lambda(i,j)P(r(\mathbf{X})=j|Y=i)P(Y=i). \end{aligned} \tag{4.4}$$

Observe por (**??**) que $R(r,i)$ é a perda esperada na alocação dos objetos da classe C_i enquanto que, por (**??**), $R(r)$ é a perda total esperada em todo o processo de alocação empregando o classificador r.

No caso particular da função de perda $0-1$, temos que

$$R(r,i) = \sum_{i\neq j=1}^{M} P(r(\mathbf{X}) = j|Y = i) \tag{4.5}$$

e

$$R(r) = \sum_{i=1}^{M} \sum_{i\neq j=1}^{M} P(r(\mathbf{X}) = j|Y = i)P(Y = i). \tag{4.6}$$

Logo, para a função de perda $0-1$, $R(r,i)$ é a probabilidade de classificação errada dos objetos da Classe C_i e $R(r)$ é a probabilidade total de classificação errada do classificador r, também denominado de *erro de classificação de r*.

Temos como objetivo agora construir um classificador que minimize o risco total. Definimos, então:

Definição 4.3. *O classificador de Bayes é definido por:*

$$r^*(\mathbf{x}) = k \quad se \quad \sum_{i=1}^{M} \lambda(i,k) f(\mathbf{x}|Y = i)P(Y = i) = \min_j \sum_{i=1}^{M} \lambda(i,j) f(\mathbf{x}|Y = i)P(Y = i) \tag{4.7}$$

no caso de o mínimo ocorrer para mais de uma classe, o objeto é alocado em qualquer uma das classes que o atingirem.

Observe que as expressões para as probabilidades *a posteriori* das classes, em (**??**), tem o mesmo denominador. Para todas as classes a regra (**??**) pode ser estabelecida de maneira equivalente como

$$r^*(\mathbf{x}) = k \quad \text{se} \quad \sum_{i=1}^{M} \lambda(i,k)P(Y = i|\mathbf{x}) = \min_j \sum_{i=1}^{M} \lambda(i,j)P(Y = i|\mathbf{x}) \tag{4.8}$$

O teorema a seguir estabelece a principal propriedade para o classificador em (**??**) e (**??**).

Teorema 4.1. *Para uma dada função de perda $\lambda(\cdot,\cdot)$, o classificador r^* minimiza o risco total, ou seja, $R(r^*) \leq R(r)$ para qualquer classificador r.*
Para a prova do Teorema **??** ver Pereira [22].

Empregando função de perda $0-1$, temos que

$$\sum_{i=1}^{M} \lambda(i,k)P(Y = i|\mathbf{x}) = \sum_{k\neq i=1}^{M} P(Y = i|\mathbf{x}) = 1 - P(Y = k|\mathbf{x}), \tag{4.9}$$

De (**??**) vemos que teremos o mínimo se tomarmos a classe C_k para a qual $P(Y = k|\mathbf{x}) = f(\mathbf{x}|Y = k)P(Y = k)$ é um máximo, isto é, a classe com a *maior probabilidade a posteriori*. Portanto, o classificador de Bayes com função de perda $0-1$ pode ser expresso como

$$r^*(\mathbf{x}) = k \quad \text{se} \quad P(Y = k|\mathbf{x}) = \max_j P(Y = j|\mathbf{x}) \tag{4.10}$$

O classificador ótimo r^* também é denominado *Regra de Bayes de Mínimo Risco*. Em teoria, o risco $e^* = R(r^*)$, denominado *Risco de Bayes*, pode ser determinado se conhecermos as probabilidades $P(Y = j)$ e as distribuições $f_j(\cdot|Y = j)$, para $j = 1,2,3,...,M$. O valor do risco de Bayes serve como referência para comparação de classificadores, pois ele é o menor valor que pode ser atingido por qualquer classificador. No caso da função de perda $0-1$, e^* é equivalente ao erro de classificação de r^* (Para mais detalhes ver Ripley [24]).

No entanto, quando enfrentamos os problemas reais, as probabilidades *a priori* $P(Y = j)$ e as densidades condicionais das classes $f_j(\cdot|Y = j)$ são desconhecidas, impossibilitando a construção do classificador de Bayes. O procedimento é, então, estimar essas quantidades com o objetivo de obter uma aproximação empírica para r^*. Neste trabalho será considerada a estimação de r^* por meio da modelagem das densidades condicionais.

Em muitos problemas reais, o procedimento amostral empregado para obtenção do conjunto de treinamento não contém informação sobre as probabilidades *a priori*. Por exemplo, às vezes o número de observações por classe é fixado previamente ou as observações são obtidas em amostragem por conveniência. Em tais situações não é possível estimar as probabilidades *a priori* $P(Y = j)$.

Nestes casos, com função de perda $(0-1)$, podemos empregar o classificador de Bayes considerando $P(Y = 1) = P(Y = 2) = ... = P(Y = M)$, que pode ser interpretada como sendo *priori não informativa*, o que leva a forma para as probabilidades *a posteriori* de (**??**) dadas por

$$P(C_j|\mathbf{x}) = P(Y = j|\mathbf{X} = \mathbf{x}) = \frac{f(\mathbf{x}|Y = j)}{\sum_{l=1}^{M} f(\mathbf{x}|Y = l)}, \quad j = 1,...,M \tag{4.11}$$

Em (**??**), na escolha do máximo $P(Y = j|\mathbf{x})$, vemos uma estrutura de máxima verossimilhança e, dessa maneira, o classificador

$$r^*(\mathbf{x}) = k \quad \text{se} \quad f(\mathbf{x}|Y = k) = \max_j f(\mathbf{x}|Y = j) \tag{4.12}$$

é denominado *classificador de máxima verossimilhança*.

Como já mencionamos, nas aplicações com problemas reais, as distribuições condicionais $f(\cdot|Y=j)$ e as probabilidades *a priori* $P(Y=j)$ são desconhecidas e devem ser estimadas com base no conjunto de treinamento. No caso de $P(Y=j)$, nem sempre isso é possível. A abordagem para estimação das distribuições $f(\cdot|Y=j)$ pode ser paramétrica, onde consideramos modelos paramétricos específicos e estimamos seus parâmetros. Uma abordagem não paramétrica também pode ser adotada, onde para as $f(\cdot|Y=j)$ são consideradas estimadores não paramétricos de densidades (ver por exemplo, McLachlan [20, Capítulo. 9]). Outra abordagem, com o emprego da *regressão logística*, pode ser adotada, em que as probabilidades $P(Y=j|\mathbf{x})$ são estimadas diretamente sem estimar as condicionais $f(\cdot|Y=j)$ (ver por exemplo, Hastie *et al.* [10, Cap. 4]).

Na seção seguinte, apresentaremos uma abordagem paramétrica bastante usual na implementação do classificador de Bayes, quando adotamos a distribuição normal multivariada para modelar as distribuições condicionais nas classes.

4.2.1 Classificação com Modelos Normais

Para as classes C_j, $j=1,2,3,..,M$, as densidades $f(\cdot|Y=j)$ são modeladas por distribuições normais multivariadas com vetor de parâmetros $\boldsymbol{\theta}_j=(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)$, denotada por $N(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)$, ou seja,

$$f_j(\mathbf{x};\boldsymbol{\theta}_j)=f(\mathbf{x}|Y=j)=\frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_j|^{\frac{1}{2}}}\exp\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_j)'\boldsymbol{\Sigma}_j^{-1}(\mathbf{x}-\boldsymbol{\mu}_j)\} \qquad (4.13)$$

onde $\boldsymbol{\mu}_j$ é o vetor de médias e $\boldsymbol{\Sigma}_j$ é a matriz de covariâncias não singular pois, a singularidade implicaria que as observações da classe não estão num espaço $\mathbb{R}^p$, ou seja, satisfazem a uma ou mais restrições lineares.

Para determinar uma forma explícita do classificador r^*, aplicamos a função logarítmo em (**??**) e, com algumas manipulações algébricas, obtemos

$$\ln\{f_k(\mathbf{x})P(Y=k)\}=-\frac{d}{2}\ln(2\pi)-\frac{1}{2}ln|\boldsymbol{\Sigma}_k|-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)'\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)+\ln P(Y=k). \quad (4.14)$$

Podemos considerar para (**??**) duas situações: uma em que as matrizes de covariâncias das classes são consideradas iguais e outra em que essas matrizes são supostas distintas.

No caso em que $\Sigma_j = \Sigma$, $j = 1,2,3,...,M$, modelo homocedástico, expandindo-se a forma quadrática em (**??**), multiplicando por -2 e desprezando-se os termos que são constantes para todas as classes, obtemos a quantidade

$$d_k^L(\mathbf{x}) = (\mathbf{x} - \boldsymbol{\mu}_k)' \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) - 2 \ln P(Y = k) \tag{4.15}$$

Empegando (**??**), com a função de perda $0-1$, o classificador de Bayes fica como

$$r^*(\mathbf{x}) = k \quad \text{se} \quad d_k^L(\mathbf{x}) = \min_j d_j^L(\mathbf{x}) \tag{4.16}$$

Devido ao fato da função $d_k^L(\mathbf{x})$ ser linear em $\mathbf{x}$, essa regra é denominada *Análise Discriminante Linear* (ADL).

O primeiro termo no segundo membro em (**??**) é, por definição, a distância de Mahalanobis ao quadrado, entre $\mathbf{x}$ e $\boldsymbol{\mu}_k$. Se as probabilidades *a priori* forem iguais para todas as classes, então, para um objeto com observação $\mathbf{x}$, a regra seleciona a classe cujo vetor de médias é o mais próximo de $\mathbf{x}$ em termos da distância de Mahalanobis. Vemos também que, se a matriz Σ for proporcional a matriz identidade, essa proximidade pode ser mensurada em termos da distância Euclidiana (Pereira [22]).

Para o caso em que as matrizes são distintas, modelo heterocedástico, com as manipulações algébricas mencionadas, obtemos a quantidade

$$d_k^Q(\mathbf{x}) = \ln |\Sigma_k| + \frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)' \Sigma_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) - 2 \ln P(Y = k), \tag{4.17}$$

e o classificador de Bayes, com função de perda $0-1$, é dado por

$$r^*(\mathbf{x}) = k \quad \text{se} \quad d_k^Q(\mathbf{x}) = \min_j d_j^Q(\mathbf{x}) \tag{4.18}$$

O classificador em (**??**) é denominado *Análise Discriminante Quadrática* (ADQ), devido a $d_k^Q(\mathbf{x})$ ser uma função quadrática em $\mathbf{x}$.

Para o uso dessas regras, faz-se necessário estimar os parâmetros $\boldsymbol{\theta}_j = (\boldsymbol{\mu}_j, \Sigma_j)$ e as probabilidades $P(Y = j)$ que, como mencionado, as estimativas são obtidas empregando-se as observações do conjunto de treinamento. Em geral, são empregados os estimadores

de Máxima Verossimilhança, dados por

$$\widehat{P}(C_j) = \frac{n_j}{n}, \tag{4.19}$$

$$\widehat{\boldsymbol{\mu}}_j = \frac{1}{n}\sum_{i=1}^{n_j}\mathbf{x}_{j,i} \quad \text{e} \tag{4.20}$$

$$\widehat{\boldsymbol{\Sigma}}_j = \frac{1}{n_j-1}\sum_{i=1}^{n_j}(\mathbf{x}_{j,i}-\widehat{\boldsymbol{\mu}}_j)(\mathbf{x}_{j,i}-\widehat{\boldsymbol{\mu}}_j)^T, \tag{4.21}$$

Observe que o estimador de $\widehat{P}(C_j)$ é a proporção de observações no conjunto de treinamento que pertencem à classe C_j. Note que, o denominador de $\widehat{\boldsymbol{\Sigma}}_j$, n_j-1, corrige o vício desse estimador. E ainda, $\widehat{\boldsymbol{\mu}}_j$ e $\widehat{\boldsymbol{\Sigma}}_j$ são, respectivamente, o *vetor de médias amostrais* e a *matriz de covariâncias amostrais* para a classe j, $j = 1,2,3,...,M$. Para mais detalhes, ver Johnson & Wichern [12].

4.2.2 Análise Discriminante empregando MF

Agora, nesta abordagem, consideramos cada classe C_j, $j = 1,2,3,..,M$, com densidade $f_j(\cdot|Y=j)$, tal que $f_j(\cdot|Y=j)$ é uma mistura finita de densidades pertencentes à Família Normal Assimétrica Independente. No trabalho, foram consideradas as densidades normal assimétrica multivariada e t assimétrica multivariada. Portanto, as classes foram modelas por

$$f_j(x;\Theta_j) = \sum_{l=1}^{g_j} p_l\, NA_p(\mathbf{x}|\boldsymbol{\mu}_l^j, \boldsymbol{\Sigma}_l^j,\ \boldsymbol{\lambda}_l^j) \tag{4.22}$$

e

$$f_j(x;\Theta_j) = \sum_{l=1}^{g_j} p_l\, tA_p(\mathbf{y}|\boldsymbol{\mu}_l^j, \boldsymbol{\Sigma}_l^j,\ \boldsymbol{\lambda}_l^j, \nu_l^j), \tag{4.23}$$

para $j = 1,2,...,M$, denotados, respectivamente, por MFNA (mistura finita de normais assimétricas) e MFtA (mistura finita de t assimétricas). Para fins de comparação, também foram consideradas nos experimentos as misturas finitas de normais multivariadas (MFN) e de t multivariadas (MFt).

O objetivo é flexibilizar a modelagem nas caudas e assimetria dos dados, na tentativa de contornar problemas em que a distribuição normal não se ajusta e, de uma maneira geral, problemas em que as distribuições simétricas não comportam os dados. Com essa abordagem, espera-se considerar pontos que seriam vistos usualmente como

outliers e aproximar o quão possível ao comportamento dos dados. Segundo Cabral *et al.* [5], as misturas têm sido amplamente aplicadas em diversas áreas científicas como uma ferramenta para modelagem de heterogeneidade da população e aproximar densidades de probabilidades complicadas, apresentando assimetria, multimodalidade e caudas pesadas, sendo um método flexível para as suposições sobre a distribuição dos dados.

Portanto, a proposta neste trabalho, como já mencionado, é modelar as distribuições condicionais $f_j(\cdot|Y = j)$ por mistura finita de densidades, considerando as componentes da mistura da família normal assimétrica independente.

Capítulo 5

Mistura Finita de Densidades

Neste capítulo apresentamos a definição de mistura finita de densidades, e suas principais particularidades. Discutimos a estimação dos seus parâmetros por máxima verossimilhança, via algoritmo EM (do termo em inglês, *Expectation and Maximization*), os resultados pertinentes a esta abordagem e o critério de seleção de modelos BIC (do termo em inglês *Bayesian Information Criterion*), adotado no trabalho.

5.1 O Modelo de Mistura Finita de Densidades

Um vetor aleatório $\mathbf{X}$ tem distribuição dada por uma *mistura finita de densidades* (MFD), com g componentes, se sua densidade é da forma

$$f(\mathbf{x};\boldsymbol{\Theta}) = \sum_{j=1}^{g} p_j f_j(\mathbf{x};\boldsymbol{\theta}_j), \tag{5.1}$$

onde $p_j \geq 0$, $j = 1,...,g$, com $\sum_{j=1}^{g} p_j = 1$, são chamados *pesos da mistura*, a densidade $f_j(.;\theta_j)$ é denominada a j-ésima *componente da mistura*, indexada por um vetor de parâmetros $\boldsymbol{\theta}_j$ e $\boldsymbol{\Theta} = ((p_1,...,p_g)^T, \boldsymbol{\theta}_1^T, ..., \boldsymbol{\theta}_g^T)^T$. A Figura 1.1 apresenta um conjunto de dados ajustados por uma MFD normais univariadas, com $g = 3$. Observe que o conjunto de dados apresenta uma distribuição complexa, com multimodalidade, situação em que uma única família paramétrica de distribuições não produziria uma modelagem satisfatória.

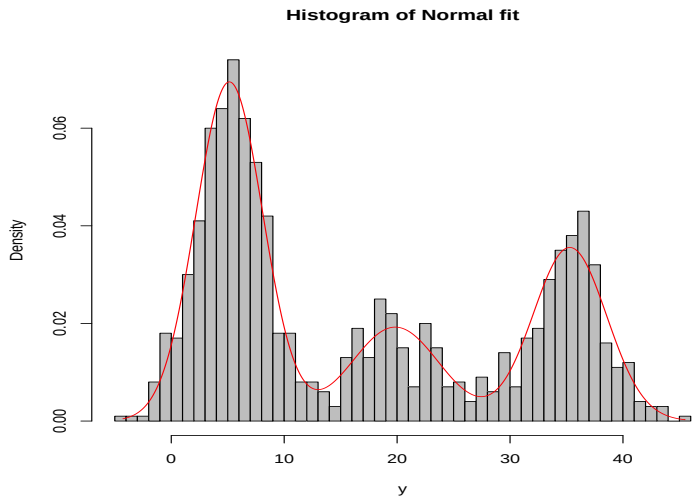


Figura 5.1: Dados ajustados por uma MFD normais univariadas, com $g = 3$.

A Figura 1.2 apresenta um conjunto de dados ajustados por uma MFD normais bivariadas, com $g = 2$. Observe que este conjunto de dados apresenta também uma distribuição complexa, com multimodalidade.

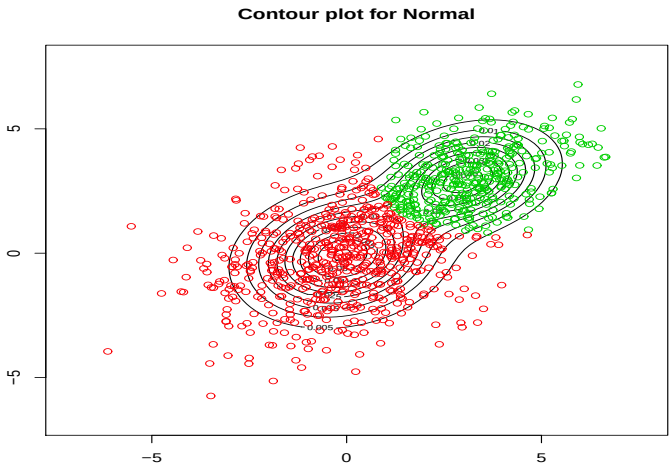


Figura 5.2: Dados ajustados por uma MFD normais bivariadas, com $g = 2$.

Para o emprego do modelo definido em (1.1), é necessário estimar os parâmetros em $\boldsymbol{\Theta}$ que, na maioria das aplicações, é implementado pelo método de máxima verossimilhança (ver, por exemplo, McLachlan & Peel [19] e Lee & McLachlan [13]). Na definição apresentada, o número de componentes g está fixado, no entanto, na prática seu valor é desconhecido sendo necessário estimá-lo a partir de dados observados para $\mathbf{X}$. Esta questão de estimar g pode ser abordada no contexto de *Seleção de Modelos*, com o emprego de uma função-critério cuja otimização indique o número de componentes adequado aos

dados.

5.1.1 Identificabilidade de MFD

Segundo McLachlan & Peel [19] a estimação de um parâmetro $\boldsymbol{\Theta}$, para uma distribuição $f(\mathbf{x}, \boldsymbol{\Theta})$, baseada nas observações de $\mathbf{X}$, só faz sentido se $\boldsymbol{\Theta}$ é identificável, isto é, possuir uma caracterização única a partir de seus parâmetros. Em geral, uma família paramétrica de funções densidades de probabilidades $f(.;\boldsymbol{\Theta})$ é dita ser identificável se valores distintos de $\boldsymbol{\Theta}$ determinam membros distintos da família de densidades. Para ver isso, considere a família de distribuição

$$\{f(\cdot;\boldsymbol{\Theta}) : \boldsymbol{\Theta} \in \Omega\},$$

onde Ω é o espaço paramétrico especificado. Então

$$f(\cdot;\boldsymbol{\Theta}) = f(\cdot;\boldsymbol{\Theta}')$$

se, e somente se,

$$\boldsymbol{\Theta} = \boldsymbol{\Theta}'.$$

A identificabilidade no caso de mistura de densidades, apresenta uma característica mais específica. Como exemplo, considere uma mistura de duas densidades normais univariadas com médias μ_1 e μ_2, e variâncias iguais a 1. Temos

$$f(x;\boldsymbol{\Theta}) = \frac{p_1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x-\mu_1)^2\right] + \frac{p_2}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x-\mu_2)^2\right].$$

onde $\boldsymbol{\Theta} = ((p_1, p_2)^T, \mu_1^T, \mu_2^T)^T$. Note que, com os valores dos parâmetros fixados, se permutarmos os índices em $\boldsymbol{\Theta}$ a densidade $f(.|\boldsymbol{\Theta})$ terá o mesmo valor em cada $x \in \mathbb{R}$. Portanto, para a identificabilidade no contexto de misturas é necessário mais uma condição, a de *permutabilidade*.

Seja $\mathscr{F} = \{\psi(\mathbf{x};\boldsymbol{\theta}) : \boldsymbol{\theta} \in \Omega, \mathbf{x} \in \mathbb{R}^p\}$ uma família paramétrica de densidades e

$$\begin{aligned}\mathscr{P} &= \{f(\mathbf{x};\Theta) : f(\mathbf{x};\Theta) = \sum_{j=1}^{g} p_j \psi(\mathbf{x};\boldsymbol{\theta}_j), \quad p_j \geq 0, \\ &\sum_{j=1}^{g} p_j = 1, \psi(\mathbf{x};\boldsymbol{\theta}) \in \mathscr{F}, \quad \Theta = ((p_1,...,p_g)^T, \boldsymbol{\theta}_1^T, ..., \boldsymbol{\theta}_g^T)^T\}. \end{aligned} \tag{5.2}$$

uma família de MFD. A classe $\mathscr{P}$ é dita identificável se, para quaisquer dois membros

$$f(\mathbf{x};\Theta) = \sum_{j=1}^{g} p_j \psi(\mathbf{x};\boldsymbol{\theta}_j) \quad \text{e} \quad f(\mathbf{x};\Theta') = \sum_{j=1}^{g'} p'_j \psi(\mathbf{x};\boldsymbol{\theta}'_j),$$

tem-se que $f(\mathbf{x};\Theta) = f(\mathbf{x};\Theta')$ se, e somente se, $g = g'$ e ainda podemos permutar os índices das componentes de forma que $p_j = p'_j$ e $\psi(\mathbf{x};\boldsymbol{\theta}_j) = \psi(\mathbf{x};\boldsymbol{\theta}'_j)$, com $j = 1,...,g$.

Como observado em McLachlan & Peel [19, Seção. 1.4], a falta de identificabilidade não é preocupante na estimação de máxima verossimilhança via algoritmo EM, no entanto, no contexto de estimação bayesiana pode causar problemas. Também, isto não se constitui um problema em aplicações onde o interesse principal é estimar o valor da densidade em observações específicas que, como será visto, é o caso em Análise Discriminante.

5.2 Estimação por Máxima Verossimilhança

Nesta seção descrevemos a estimação por máxima verossimilhança e discutimos algumas particularidades no contexto de MFD.

Definição 5.1. *Seja* $\mathbf{X}$ *um vetor aleatório com distribuição dada por* $f(\cdot;\Theta)$*, com* $\Theta \in \Omega$*. Dado* $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ *um conjunto de n observações independentes de* $\mathbf{X}$*, a Função de Verossimilhança (FV) é definida como*

$$L(\Theta) = L(\Theta|\mathbf{x}) = \prod_{i=1}^{n} f(\mathbf{x}_i;\Theta). \tag{5.3}$$

Sendo $f(\cdot;\Theta)$ uma mistura finita de densidades, temos

$$L(\Theta) = \prod_{i=1}^{n} \{\Sigma_{j=1}^{g} p_j f_j(\mathbf{x}_i; \boldsymbol{\theta}_j)\}, \tag{5.4}$$

a função de verossimilhança para o modelo de MFD.

Definição 5.2. *Se existe um único valor* $\widehat{\Theta} \in \Omega$ *que maximiza a Função de Verossimilhança, então* $\widehat{\Theta}$ *é denominado* Estimador de Máxima Verossimilhança *(EMV) de* Θ.

Devido ao problema da falta de identificabilidade, existem situações em que a FV atinge máximo local para diferentes valores de Θ, por esse motivo, as equações de verossimilhança tem várias raízes. Como observado em McLachlan & Krishnan [21], esse problema pode ser grave quando há interesse na estimação dos valores específicos para os parâmetros envolvidos. Por outro lado, se o objetivo é apenas estimar o valor da densidade em pontos de interesse que, como já mencionado, é o caso em Análise Discriminante, isto não se constitui em problema (ver Ripley [24, Cap. 6]).

Outra situação particular é que no modelo de misturas finitas a FV pode não ser limitada superiormente (ver McLachlan & Peel [19, Seção 1.13]), desta forma, o EMV pode não existir. Como observado em Ripley [24, Seção 6.4], em muitos problemas reais, por exemplo, nas aplicações em Análise Discriminante, é necessário somente obter uma "boa aproximação" para o valor estimado da densidade nos pontos de interesse, portanto, é possível obter uma estimativa para Θ que permita o emprego das MFD como modelos para aproximarem as distribuições envolvidas no problema.

Na prática, a determinação dos EMV para MFD envolvem várias dificuldades devido à complexidade da dependência da FV nos parâmetros. As equações de máxima verossimilhança não são lineares, não sendo possível obter soluções analíticas, portanto, é necessário o emprego de métodos numéricos de maximização de funções (Pereira [22]). Para esse fim, segundo McLachlan & Krishnan [21] e Lee & McLachlan [14], o algoritmo EM é o mais empregado nesse contexto.

5.3 Algoritmo EM para Modelos de Misturas Finitas

Nesta seção descrevemos o algoritmo EM para mistura finita de densidades, sendo necessário definirmos inicialmente a questão da estrutura de dados incompletos que é empregada pelo algoritmo.

Seja $\mathbf{X}_1, ...\mathbf{X}_n$ uma amostra aleatória de $\mathbf{X}$, com distribuição dada por (1.1). Para $i = 1, ..., n$, seja $\mathbf{Z}_i$ um vetor aleatório, com $\mathbf{Z}_i = (Z_{i1}, ..., Z_{ig})$, cujas componentes são definidas como variáveis indicadoras da forma

$$Z_{ij} = \begin{cases} 1, & \text{se } \mathbf{X}_i \sim f_j(\mathbf{x}; \boldsymbol{\theta}_j); \\ 0, & \text{se } \mathbf{X}_i \sim f_l(\mathbf{x}; \boldsymbol{\theta}_l),\ l \neq j. \end{cases}$$

Podemos interpretar essa variável Z_{ij} como uma variável latente, não observável, associada ao vetor $\mathbf{X}_i$ e indicando qual componente da mistura descreve sua distribuição.

Neste contexto, os valores observados $\mathbf{x}_1, ..., \mathbf{x}_n$ de $\mathbf{X}_1, ...\mathbf{X}_n$, são considerados *dados incompletos*. Denotando por $\mathbf{z}_1, ...\mathbf{z}_n$ os valores para $\mathbf{Z}_1, ...\mathbf{Z}_n$, os vetores de *dados completos* são, portanto,

$$\mathbf{y}_i = (\mathbf{x}_i, \mathbf{z}_i), \quad i = 1, ..., n.$$

Assumimos então que o vetor $\mathbf{Z}_i$ tem distribuição multinomial, considerando uma retirada em g categorias, com probabilidades $p_1, ..., p_g$, isto é, formam uma amostra aleatória

$$\mathbf{Z}_1, ..., \mathbf{Z}_n \sim Multi_g(1; p_1, ..., p_g),$$

portanto a distribuição de $\mathbf{Z}_i$ é da forma

$$f(\mathbf{z}_i; \boldsymbol{\alpha}) = \prod_{j=1}^{g} p_j^{z_{ij}}, \text{ onde } \boldsymbol{\alpha} = (p_1, ..., p_g). \tag{5.5}$$

Pela construção acima, o j-ésimo peso da mistura (p_j) pode ser interpretado como a probabilidade *a priori* da i-ésima observação provir da j-ésima componente da mistura, com $i = 1, ..., n$.

Pelas suposições estabelecidas, a distribuição conjunta para os dados completos $\mathbf{y}_i$ é da forma

$$
\begin{aligned}
f(\mathbf{y}_i;\Theta) &= f(\mathbf{x}_i|\mathbf{z}_i;\Theta)\,f(\mathbf{z}_i;\boldsymbol{\alpha}) \\
&= \prod_{j=1}^{g} f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)^{z_{ij}}\,f(\mathbf{z}_i;\boldsymbol{\alpha}) \\
&= \prod_{j=1}^{g} f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)^{z_{ij}}\,p_j^{z_{ij}}. \qquad (5.6)
\end{aligned}
$$

Para o desenvolvimento do EM, construimos a FV dos dados completos $\mathbf{y} = (\mathbf{y}_1, ..., \mathbf{y}_n)$, dada por

$$
L(\Theta) = L(\Theta|\mathbf{y}) = \prod_{i=1}^{n} f(\mathbf{y}_i;\Theta) = \prod_{i=1}^{n}\prod_{j=1}^{g} f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)^{z_{ij}}\,p_j^{z_{ij}}, \qquad (5.7)
$$

e o seu logaritmo (na base e) dado por

$$
\begin{aligned}
l_c(\Theta) &= \log L(\Theta) = \sum_{i=1}^{n}\sum_{j=1}^{g} z_{ij}\{\log p_j + \log f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)\} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{g} z_{ij}\log p_j + \sum_{i=1}^{n}\sum_{j=1}^{g} z_{ij}\log f_j(\mathbf{x}_i;\boldsymbol{\theta}_j). \qquad (5.8)
\end{aligned}
$$

Para a construção do algoritmo EM determinamos a função Q

$$
Q(\Theta|\Theta^{(k)}) = E\{l_c(\Theta)|\mathbf{x};\Theta^{(k)}\}, \qquad (5.9)
$$

onde $\mathbf{x} = (\mathbf{x}_1, ..., \mathbf{x}_n)$ e a notação (k) sobrescrito informa que essa é a estimação do parâmetro obtida no instante k do algoritmo. Observe que $Q(\Theta|\Theta^{(k)})$ é a esperança calculada usando $\boldsymbol{\theta}^{(k)}$ como valor para $\boldsymbol{\theta}$. O algoritmo, então, se desenvolve em dois passos:

1. Passo E: Determinar $Q(\Theta|\Theta^{(k)})$;
2. Passo M: Escolher $\Theta^{(k+1)}$ maximizando $Q(\Theta|\Theta^{(k)})$.

Uma propriedade importante do algoritmo EM é que a forma como as estimativas $\Theta^{(k)}$ são determinadas garante que $Q(\Theta^{(k+1)}|\Theta^{(k)}) \geq Q(\Theta^{(k)}|\Theta^{(k)})$ (ver, por exemplo

Dempster *et al.* [6]). É demonstrado (ver, por exemplo, Pereira [22]) que este fato implica em

$$l(\boldsymbol{\Theta}^{(k+1)}) \geq l(\boldsymbol{\Theta}^{(k)}), \tag{5.10}$$

portanto, as aproximações $l(\boldsymbol{\Theta}^{(k)})$ obtidas pelo algoritmo EM geram uma sequência $\{l(\boldsymbol{\Theta}^{(k)})\}$ não-decrescente.

Na implementação do algoritmo, os passos E e M são repetidos alternadamente até que um critério de convergência adequado seja atingido. Várias propostas para critérios de parada são discutidas na literatura (ver, por exemplo, McLachlan & Peel [19] e McLachlan & Krishnan [21]). Neste trabalho, o critério de convergência adotado é da forma

$$\left| \frac{l(\boldsymbol{\Theta}^{(k+1)})}{l(\boldsymbol{\Theta}^{(k)})} - 1 \right| < c, \tag{5.11}$$

com c suficientemente pequeno.

Para a expressão (1.9), obtemos

$$Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) = \sum_{i=1}^{n}\sum_{j=1}^{g} E[Z_{ij}|\mathbf{x};\boldsymbol{\Theta}^{(k)}] \ln p_j + \sum_{i=1}^{n}\sum_{j=1}^{g} E[Z_{ij}|\mathbf{x};\boldsymbol{\Theta}^{(k)}] \ln f_j(\mathbf{x}_i;\boldsymbol{\theta}_j^{(k)}), \tag{5.12}$$

onde $\boldsymbol{\Theta}^{(k)} = ((p_1^{(k)},...,p_g^{(k)})^T, \boldsymbol{\theta}_1^{T(k)},...,\boldsymbol{\theta}_g^{T(k)})^T$.

De (1.12), vemos que é necessário determinar

$$\tau_{ij}^{(k)} \stackrel{\text{def}}{=} E[Z_{ij}|\mathbf{x};\boldsymbol{\Theta}^{(k)}] = P[Z_{ij}=1|\mathbf{x};\boldsymbol{\Theta}^{(k)}] = P[Z_{ij}=1|\mathbf{x}_i,\boldsymbol{\Theta}^{(k)}], \tag{5.13}$$

Considerando a distribuição dada em (1.5), temos que

$$P[Z_{ij}=1|\boldsymbol{\Theta}^{(k)}] = P[Z_{ij}=1, Z_{il}=0 \forall l \neq j|\boldsymbol{\Theta}^{(k)}] = p_j^{(k)}. \tag{5.14}$$

Usando (1.6), vemos que

$$f(\mathbf{x}_i|\mathbf{Z}_i,\boldsymbol{\Theta}^{(k)}) = \prod_{j=1}^{g} (f_j(\mathbf{x}_i;\boldsymbol{\theta}_j^{(k)}))^{z_{ij}}, \tag{5.15}$$

e, portanto, temos que

$$f(\mathbf{x}_i|Z_{ij}=1;\boldsymbol{\Theta}^{(k)}) = f_j(\mathbf{x}_i;\boldsymbol{\theta}_j^{(k)}). \tag{5.16}$$

Empregando o Teorema de Bayes, vemos que

$$\tau_{ij}^{(k)} = P[Z_{ij}=1|\mathbf{x}_i,\boldsymbol{\Theta}^{(k)}] = \frac{P[Z_{ij}=1|\boldsymbol{\Theta}^{(k)}]\, f(\mathbf{x}_i|Z_{ij}=1,\boldsymbol{\Theta}^{(k)})}{f(\mathbf{x}_i;\boldsymbol{\Theta}^{(k)})}. \tag{5.17}$$

Usando (1.1), (1.14) e (1.16) em (1.17), obtemos

$$\tau_{ij}^{(k)} = \frac{p_j^{(k)} f_j(\mathbf{x}_i;\boldsymbol{\theta}_j^{(k)})}{\sum_{t=1}^{g} p_t^{(k)} f_t(\mathbf{x}_i;\boldsymbol{\theta}_t^{(k)})}. \tag{5.18}$$

Da expressão acima, vemos que $\tau_{ij}^{(k)}$ é uma estimativa da probabilidade da observação $\mathbf{x}_i$ provir da componente $f_j(\cdot;\boldsymbol{\theta}_j^{(k)})$ da mistura, com base em uma dada estimativa $\boldsymbol{\Theta}^{(k)}$ do vetor de parâmetros $\boldsymbol{\Theta}$.

Agora, usando (1.18) em (1.12), podemos escrever

$$\begin{aligned} Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) &= \sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(k)} \log p_j + \sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(k)} \log f_j(\mathbf{x}_i;\boldsymbol{\theta}_j) \\ &= Q_1(\boldsymbol{\alpha}) + Q_2(\boldsymbol{\theta}), \end{aligned} \tag{5.19}$$

onde $\boldsymbol{\alpha} = (p_1,...,p_g)$ e $\boldsymbol{\theta} = (\boldsymbol{\theta}_1,...,\boldsymbol{\theta}_g)$,

$$Q_1(\boldsymbol{\alpha}) = \sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(g)} \log p_j \tag{5.20}$$

e

$$Q_2(\boldsymbol{\theta}) = \sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(k)} \log f_j(\mathbf{x}_i;\boldsymbol{\theta}_j). \tag{5.21}$$

Como pode ser visto na expressão (1.19), o problema de maximização de $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)})$ é considerado, separadamente, para $\boldsymbol{\alpha}$ e $\boldsymbol{\theta}$, ou seja, dois problemas que envolvem

$$\frac{\partial Q_1(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}} = \mathbf{0} \quad \text{e} \quad \frac{\partial Q_2(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0}.$$

Note que o primeiro problema de maximização tem uma solução única que é explicitamente determinada, independente da forma funcional das componentes da mistura. Considerando as restrições sobre $\boldsymbol{\alpha}$, os p_j são não negativos e somam 1, a solução é obtida com o emprego de multiplicadores de Lagrange, sendo dada por (ver Dempster *et al.* [6])

$$p_j^{(k+1)} = \frac{1}{n}\sum_{i=1}^{n} \tau_{ij}^{(k)}, \qquad j = 1,2,...,g. \tag{5.22}$$

Dessa maneira, determinamos $\tau_{ij}^{(k)}$ por meio de (1.18) e as aproximações para $\boldsymbol{\alpha}^{(k+1)}$ por (1.22).

No segundo problema de maximização, com relação à $\boldsymbol{\theta}$, vemos que a solução depende da forma funcional das componentes $f_j(\mathbf{x}_i;\boldsymbol{\theta}_j)$, pois devemos solucionar

$$\sum_{i=1}^{n}\sum_{j=1}^{g} \tau_{ij}^{(k)} \frac{\partial}{\partial \boldsymbol{\theta}} \ln f_j(\mathbf{x}_i;\boldsymbol{\theta}_j) = \mathbf{0}. \tag{5.23}$$

Para a equação (1.23) não há garantia de solução única e bem definida. Isto será discutido no Capítulo 3, no contexto de mistura finita de densidades com as componentes na família de *distribuições normais assimétricas independentes*.

5.4 Estimação do número de componentes

A abordagem da estimação do número de componentes consiste em considerar uma função-critério cuja otimização indique o número de componentes do modelo adequado aos dados. Uma função-critério, em geral, seleciona o número $\widehat{g}$ de componentes da forma

$$\widehat{g} = arg\,min_g\{C(\widehat{\boldsymbol{\Theta}}_{(g)}), g = g_{min},...,g_{max}\},$$

onde $C(\widehat{\boldsymbol{\Theta}}_{(g)})$ é o valor da função-critério para o modelo estimado com dimensão g, no caso de mistura finita de densidades, o modelo com g componentes. Existem várias propostas na literatura para a função-critério (ver, por exemplo, McLachlan & Peel [19] e Hastie *et al.* [10]). Segundo Andrews & McNicholas [1], um dos critérios de seleção de modelos mais empregado é o Critério de Informação Bayesiano (ver, também, Wang & Hu [27], Wang *et al.* [28] e Fraley & Raftery [8]), que denotaremos por BIC, devido ao termo em inglês *Bayesian Information Criterion*, que será adotado neste trabalho.

O BIC é baseado na teoria Bayesiana de seleção de modelos, onde são considerados vários possíveis modelos, com suas probabilidades *a priori*, objetivando selecionar

o modelo com a maior probabilidade "a posteriori", dadas as observações. Considere os modelos $M_1,...,M_G$ e suas respectivas probabilidades *a priori* $P(M_g), g = 1,2,3,...,G$. Pelo Teorema de Bayes, "a posteriori" de M_g dado $\mathbf{x} = (\mathbf{x}_1,...,\mathbf{x}_n)$ é

$$P(M_g|\mathbf{x}) = \frac{f(\mathbf{x}|M_g)P(M_g)}{\sum_{t=1}^{G} f(\mathbf{x}|M_t)P(M_t)}. \tag{5.24}$$

Para essa expressão, é necessário determinar $f(\mathbf{x}|M_g)$, que denota a distribuição de $\mathbf{x}$ segundo o modelo M_g. Como os parâmetros são geralmente desconhecidos, obtemos (1.24) integrando sobre o espaço paramétrico Ω,

$$f(\mathbf{x}|M_g) = \int L(\mathbf{x}|\boldsymbol{\Theta}, M_g)\pi(\boldsymbol{\Theta}_{(g)}|M_g)d\boldsymbol{\Theta}_{(g)}, \tag{5.25}$$

onde $\pi(\boldsymbol{\Theta}_{(g)}|M_g)$ é a distribuição *a priori* para $\boldsymbol{\Theta}_{(g)}$ e $L(\mathbf{x}|\boldsymbol{\Theta}, M_g)$ é a função de verossimilhança para o modelo M_g com vetor de parâmetros $\boldsymbol{\Theta}_{(g)}$.

Se as probabilidades $P(M_g)$ em (1.24) forem iguais, o procedimento seleciona o modelo que apresentar maior valor para $f(\mathbf{x}|M_g)$, denominada de *verossimilhança integrada*. Em todos os casos, é necessário obter uma aproximação para a integral (1.25), o desenvolvimento para obter esta aproximação está descrito em Basso [4], sendo o resultado da forma

$$BIC(g) = -2l(\widehat{\boldsymbol{\Theta}}_g) + \upsilon(g)\log n, \tag{5.26}$$

onde $l(\widehat{\boldsymbol{\Theta}}_{(k)})$ é o logaritmo da função de máxima verossimilhança para o modelo com vetor de parâmetros estimados $\widehat{\boldsymbol{\Theta}}_{(g)}$, e $\upsilon(g)$ é o número de parâmetros livres no modelo de dimensão g. Portanto, o procedimento seleciona o modelo com o número de componentes que apresenta o menor valor para o BIC.

Na literatura, é mencionado que o BIC assintoticamente tende a selecionar o modelo de dimensão correta, por isso é denominado *consistente em ordem* (ver referências em Pereira [22]). Apesar de não ter sido desenvolvido para modelos de misturas, na prática este critério tem apresentado resultados satisfatórios para selecionar o número de componentes da mistura ao aproximar densidades desconhecidas empregando misturas finitas de densidades (ver Pereira [22], Basso [4] e as referências nestes trabalhos).

poderíamos até adotar uma fronteira de decisão linear admitindo um pequeno erro de classificação.

As observações em cada classe foram geradas por MFN. Para a Classe 1 foi empregada uma mistura de cinco componentes com densidade normal, com parâmetros

$$
\begin{aligned}
\alpha_{11} &= \alpha_{12} = \alpha_{13} = \alpha_{14} = \alpha_{15} = 0,2; \\
\boldsymbol{\mu}_{11} &= \begin{pmatrix} 5 \\ 2 \end{pmatrix}, \quad \boldsymbol{\mu}_{12} = \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \quad \boldsymbol{\mu}_{13} = \begin{pmatrix} 2 \\ 5 \end{pmatrix}, \quad \boldsymbol{\mu}_{14} = \begin{pmatrix} 3 \\ 7 \end{pmatrix}, \quad \boldsymbol{\mu}_{15} = \begin{pmatrix} 5 \\ 8 \end{pmatrix}; \\
\boldsymbol{\Sigma}_{11} &= \begin{pmatrix} 1 & -0,5 \\ -0,5 & 1 \end{pmatrix}, \quad \boldsymbol{\Sigma}_{12} = \boldsymbol{\Sigma}_{13} = \boldsymbol{\Sigma}_{14} = \boldsymbol{\Sigma}_{15} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.
\end{aligned}
$$

Para a Classe 2 foi empregada igualmente uma mistura de cinco componentes normais, com parâmetros

$$
\begin{aligned}
\alpha_{21} &= \alpha_{22} = \alpha_{23} = \alpha_{24} = \alpha_{25} = 0,2; \\
\boldsymbol{\mu}_{21} &= \begin{pmatrix} 9 \\ 5 \end{pmatrix}, \quad \boldsymbol{\mu}_{22} = \begin{pmatrix} 11 \\ 6 \end{pmatrix}, \quad \boldsymbol{\mu}_{23} = \begin{pmatrix} 12 \\ 8 \end{pmatrix}, \quad \boldsymbol{\mu}_{24} = \begin{pmatrix} 11 \\ 10 \end{pmatrix}, \quad \boldsymbol{\mu}_{25} = \begin{pmatrix} 9 \\ 11 \end{pmatrix}; \\
\boldsymbol{\Sigma}_{21} &= \boldsymbol{\Sigma}_{22} = \boldsymbol{\Sigma}_{23} = \boldsymbol{\Sigma}_{24} = \boldsymbol{\Sigma}_{25} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.
\end{aligned}
$$

Na Tabela **??** são apresentados os resultados obtidos com a classificação (em porcentagem) do conjunto de teste, segundo o tamanho dos conjuntos de treinamento, para cada uma das distribuições.

n_{tre}	MFN	MFNA	MFt	MFtA
200	0,44(0,26) [0,09;1,00]	0,45(0,30) [0,09;1,20]	0,44(0,26) [0,10;1,01]	0,44(0,29) [0,00;1,10]
300	0,39(0,23) [0,00;0,91]	0,39(0,22) [0,00;0,89]	0,40(0,24) [0,00;0,96]	0,40(0,24) [0,00;0,90]
500	0,32(0,19) [0,00;0,80]	0,35(0,21) [0,00;0,81]	0,32(0,19) [0,00;0,79]	0,34(0,20) [0,00;0,79]
1000	**0,22**(0,17) [0,00;0,70]	0,31(0,18) [0,00;0,71]	0,30(0,17) [0,00;0,70]	0,30(0,18) [0,00;0,70]

Tabela 5.1: Problema 4 - Média da Taxa de Erro (desvio-padrão) [intervalo de confiança empírico].

A maior média de TE obtida foi para a MFNA $(0,45\%)$ para $n_{tre} = 200$. Os resultados foram satisfatórios para este problema, onde as classes são bem separadas. Note ainda que para todos os modelos houve decrescimento da TE com o aumento do tamanho do conjunto de treinamento e para n_{tre} maior do que 300 os intervalos empíricos para as TE's contêm o zero. Na Figura **??**, onde apresentamos os *boxplots* das TE's para esse problema, observe que ocorre uma diminuição da dispersão com o aumento do tamanho da amostra. Note também, a maior variabilidade das TE's para $n_{tre} = 200$, o que possivelmente se verifica pelo exposto anteriormente (ver Problema 3).

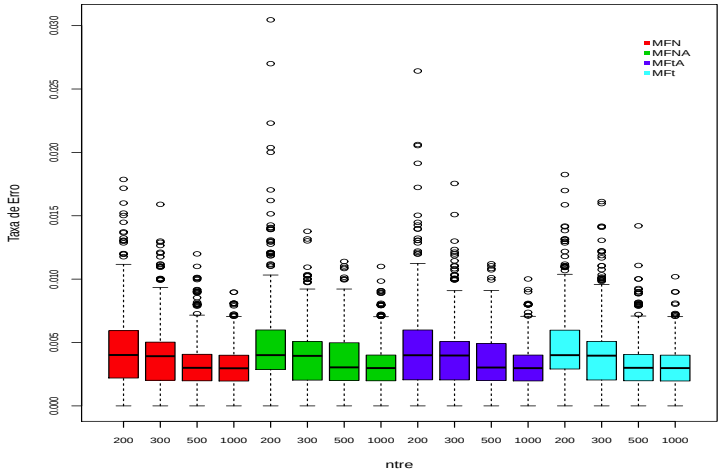


Figura 5.3: Taxas de erro do Problema 4.

Com a variação do tamanho do conjunto de treinamento e sendo feita 1000 repetições, observamos entre os resultados obtidos para a Classe 1, que os modelos selecionaram com maior frequência duas componentes e, em algumas poucas repetições, foram selecionadas 3 componentes.

Para a Classe 2 os resultados foram razoavelmente melhores, onde cada modelo, com exceção do modelo de MFtA, selecionou com maior frequência duas componentes. O modelo de MFtA apresentou um comportamento um pouco diferente dos demais nesta classe, para conjunto de treinamento de tamanhos 200 e 300 o modelo selecionou com maior frequência apenas uma componente. Aumentando o conjunto de treinamento, para n_{tre} igual a 200 e 300, o modelo selecionou com maior frequência duas componentes.

Note que, comparando com o Problema 2, simulado com o mesmo número de componentes por classe, este problema apresenta as distribuições das classes mais próximas e, provavelmente, isto contribuiu para o pequeno aumento nas TE's.

5.4.1 Problema 5

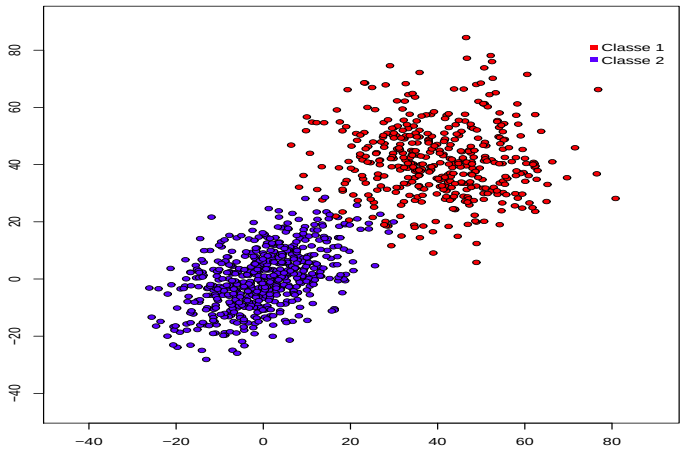


Figura 5.4: Gráfico do Problema 5.

No Problema 5, simulamos as observações de maneira que as classes estão bem separadas. A Classe 1 foi gerada com probabilidade $P_1 = 0,4$ e a Classe 2 com probabilidade $P_2 = 0,6$. Note que, se admitirmos um pequeno erro de classificação, assim como o problema anterior poderia ser adotada uma fronteira de decisão linear. Observamos na Figura **??** um exemplo com 1000 pontos simulados para este problema, que foi considerado no estudo por ser uma situação relativamente simples para modelar, porém observe que há um certa sobreposição das classes devido aos valores elevados das variâncias.

As distribuições consideradas foram normais. A Classe 1 foi gerada por uma normal, com parâmetros

$$\boldsymbol{\mu}_1 = \begin{pmatrix} 40 \\ 40 \end{pmatrix}, \qquad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 180 & 0 \\ 0 & 180 \end{pmatrix}.$$

A Classe 2 foi gerada por uma normal, com parâmetros

$$\boldsymbol{\mu}_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 100 & 50 \\ 50 & 100 \end{pmatrix}.$$

Na Tabela **??** são apresentados os resultados obtidos para este problema (em porcentagem), segundo o tamanho dos conjuntos de treinamento, para cada uma das distribuições.

n_{tre}	MFN	MFNA	MFt	MFtA
200	1,25(0,37) [0,59;2,00]	1,33(0,39) [0,64;2,14]	1,28(0,36) [0,60;2,00]	1,28(0,40) [0,60;2,11]
300	1,26(0,36) [0,59;2,02]	1,31(0,38) [0,63;2,12]	1,24(0,36) [0,52;2,00]	1,29(0,39) [0,60;2,10]
500	1,21(0,36) [0,59;1,98]	1,29(0,38) [0,60;2,08]	**1,20**(0,35) [0,51;1,99]	1,28(0,36) [0,60;2,10]
1000	1,21(0,35) [0,52;1,90]	1,30(0,37) [0,62;2,08]	**1,20**(0,35) [0,60;1,91]	1,26(0,36) [0,60;2,00]

Tabela 5.2: Problema 5 - Média da Taxa de Erro (desvio-padrão) [intervalo de confiança empírico].

Para este problema, o aumento do tamanho do conjunto de treinamento proporcionou uma diminuição da variabilidade, comportamento sugerido pelo desvio-padrão. Apesar de a média das TE's não terem aparentemente diminuído de maneira geral com o aumento do conjunto de treinamento, observe que o limite superior de cada IC obtido para TE decresceu com o aumento da amostra.

A maior média da TE obtida foi para MFNA $(1,33\%)$, com a menor amostra simulada $(n_{tre} = 200)$. O ajuste que apresentou a menor média para TE foi com MFt $(1,20\%)$ para $n_{tre} = 500$ e $n_{tre} = 1000$. Na Figura **??**, temos os *boxplots* para as TE's obtidas neste problema, onde observamos os comportamentos equivalentes dos modelos em relação às suas TE's.

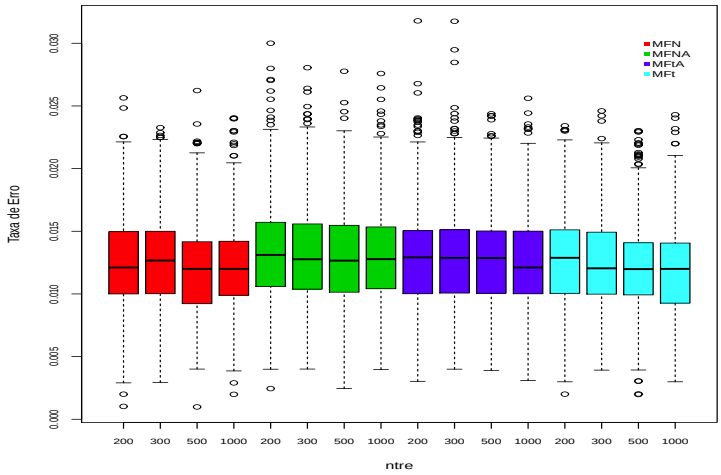


Figura 5.5: Taxas de erro do Problema 5.

Neste problema, assim como o Problema 3, cada classe foi gerada por uma distribuição normal e apesar de diferirem em complexidade, pois aqui temos uma estrutura realmente simples, o desempenho dos modelos na seleção do número de componentes foi igualmente satisfatório. Todos os modelos de misturas selecionaram apenas uma componente em cada classe.

5.4.2 **Problema** 6

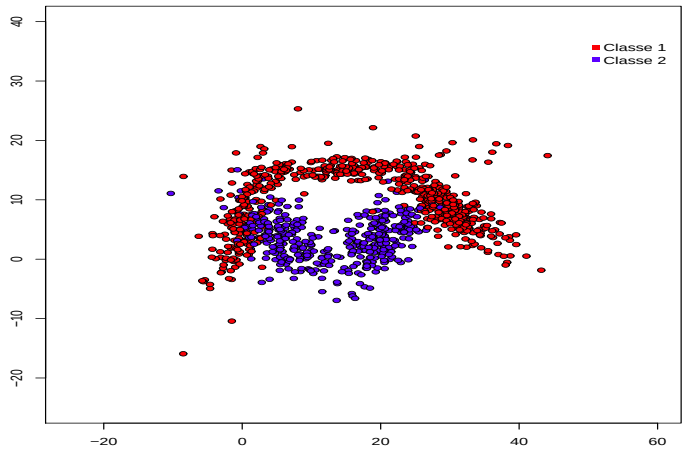


Figura 5.6: Gráfico do Problema 6.

Para o Problema 6, a Classe 1 foi gerada com probabilidade $P_1 = 0,6$ e a Classe 2 com probabilidade $P_2 = 0,4$. Na Figura **??**, foram simuladas 1000 observações para vizualizarmos a estrutura das duas classes e como é a fronteira de decisão entre elas. Observamos que a fronteira de decisão é não linear.

A Classe 1 foi simulada por uma MFt com três componentes, com parâmetros

$$\begin{aligned}
\alpha_{11} &= 0,3, \quad \boldsymbol{\mu}_{11} = \begin{pmatrix} 0 \\ 6 \end{pmatrix}, \quad \boldsymbol{\Sigma}_{11} = \begin{pmatrix} 8 & 10,74 \\ 10,74 & 30 \end{pmatrix}, \quad \nu_{11} = 10; \\
\alpha_{12} &= 0,3, \quad \boldsymbol{\mu}_{12} = \begin{pmatrix} 15 \\ 15 \end{pmatrix}, \quad \boldsymbol{\Sigma}_{12} = \begin{pmatrix} 40 & 30 \\ 5 & 2 \end{pmatrix}, \quad \nu_{12} = 10; \\
\alpha_{13} &= 0,4, \quad \boldsymbol{\mu}_{13} = \begin{pmatrix} 30 \\ 8 \end{pmatrix}, \quad \boldsymbol{\Sigma}_{13} = \begin{pmatrix} 15 & -10 \\ -10 & 10 \end{pmatrix}, \quad \nu_{13} = 10.
\end{aligned}$$

Para Classe 2 simulamos observações por uma MFt, assim como a Classe 1, mas com duas componentes, com parâmetros

$$\alpha_{21} = 0,5, \quad \boldsymbol{\mu}_{21} = \begin{pmatrix} 6 \\ 3,5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_{21} = \begin{pmatrix} 10 & -5 \\ -5 & 10 \end{pmatrix}, \quad \nu_{21} = 10;$$
$$\alpha_{22} = 0,5, \quad \boldsymbol{\mu}_{22} = \begin{pmatrix} 20 \\ 3,5 \end{pmatrix}, \quad \boldsymbol{\Sigma}_{12} = \begin{pmatrix} 10 & 5 \\ 5 & 10 \end{pmatrix}, \quad \nu_{22} = 10.$$

Na Tabela **??** são apresentados as TE's de classificação do conjunto de teste, segundo o tamanho dos conjuntos de treinamento, para cada uma das distribuições propostas para ajuste.

n_{tre}	MFN	MFNA	MFt	MFtA
200	$6,29(0,87)\,[4,69;8,11]$	$6,46(0,93)\,[4,79;8,39]$	$6,27(0,86)\,[4,64;8,05]$	$6,44(0,91)\,[4,82;8,32]$
300	$6,10(0,78)\,[4,60;7,64]$	$6,23(0,79)\,[4,75;7,86]$	$6,05(0,76)\,[4,60;7,66]$	$6,19(0,82)\,[4,67;7,91]$
500	$5,92(0,77)\,[4,37;7,37]$	$6,07(0,80)\,[4,53;7,74]$	$5,93(0,74)\,[4,48;7,38]$	$6,03(0,75)\,[4,66;7,57]$
1000	$5,87(0,74)\,[4,41;7,26]$	$5,98(0,76)\,[4,48;7,43]$	$\mathbf{5,86}(0,75)\,[4,44;7,33]$	$5,97(0,75)\,[4,50;7,33]$

Tabela 5.3: Problema 6 - Média da Taxa de Erro (desvio-padrão) [intervalo de confiança empírico].

Mais uma vez observamos que para todos os modelos houve decrescimento das médias das taxas com o aumento do conjunto de treinamento.

Para conjunto de treinamento de tamanho 200, a menor média foi obtida foi com o ajuste de MFt $(6,27\%)$. Com $n_{tre} = 300$ a menor média para TE foi para MFt $(6,05\%)$. Para $n_{tre} = 500$, a menor média observada foi para MFN $(0,5,92\%)$ e para $n_{tre} = 1000$, os ajustes tiveram resultados similares, mas em valor absoluto o modelo de MFt apresentou menor TE $(5,86\%)$. Neste problema também há uma sobreposição dos IC's obtidos para todos os modelos.

Na Figura **??**, temos os *boxplots* para as TE's obtidas neste problema, onde observamos os comportamentos equivalentes dos modelos em relação às suas TE's e, novamente, observamos uma variabilidade mais acentuada para $n_{tre} = 200$

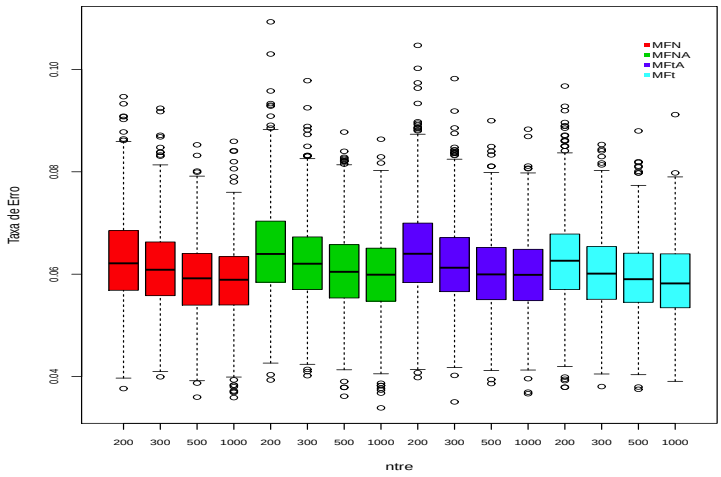


Figura 5.7: Taxas de erro do Problema 6.

Na seleção do número de componentes para a Classe 1, que foi gerada por um modelo de MFt com 3 componentes, todos os modelos apresentaram comportamento similar, selecionando com maior frequência 3 componentes. Para a Classe 2, todos os modelos selecionaram com maior frequência duas componentes.

O modelo de MFt não apresentou variações na escolha do número de componentes da mistura, como esperado, selecionou três componentes para a Classe 1 e duas componentes para a Classe 2.

5.4.3 **Problema** 7

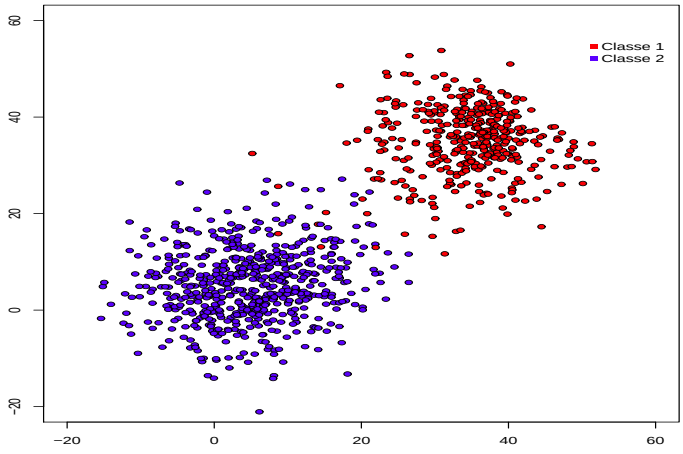


Figura 5.8: Gráfico do Problema 7.

Para o Problema 7, simulamos as observações de maneira que as classes estão bem separadas. A Classe 1 foi gerada com probabilidade $P_1 = 0,4$ e a Classe 2 com probabilidade $P_2 = 0,6$. Na Figura **??**, foram simuladas 1000 observações para vizualizarmos a estrutura das duas classes e como é a fronteira de decisão entre elas.

Para as observações da Classe 1, ajustamos um modelo com distribuição t assimétrica, com parâmetros

$$\boldsymbol{\mu}_1 = \begin{pmatrix} 40 \\ 40 \end{pmatrix}, \quad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 60 & 20 \\ 20 & 60 \end{pmatrix}, \quad \lambda_{11} = \begin{pmatrix} -2 \\ -2 \end{pmatrix}, \quad \nu = 10.$$

E para a Classe 2, simulamos observações com distribuição normal assimétrica, com parâmetros

$$\boldsymbol{\mu}_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 80 & 30 \\ 30 & 80 \end{pmatrix}, \quad \lambda_{21} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Na Tabela **??** são apresentadas as TE's de classificação do conjunto de teste, segundo o tamanho do conjunto de treinamento, para cada uma das MF ajustadas, na tentativa de modelar a distribuição de cada classe.

n_{tre}	MFN	MFNA	MFt	MFtA
200	0,88(0,30) [0,30;1,53]	0,90(0,33) [0,32;1,62]	0,89(0,32) [0,30;1,55]	0,88(0,33) [0,30;1,59]
300	0,87(0,30) [0,30;1,50]	0,88(0,30) [0,39;1,52]	0,84(0,29) [0,30;1,41]	0,84(0,30) [0,30;1,50]
500	0,86(0,29) [0,30;1,45]	0,84(0,31) [0,30;1,50]	0,83(0,29) [0,30;1,41]	**0,81**(0,28) [0,30;1,40]
1000	0,85(0,29) [0,31;1,49]	0,84(0,30) [0,30;1,50]	**0,81**(0,29) [0,30;1,43]	**0,81**(0,28) [0,30;1,41]

Tabela 5.4: Problema 7 - Média da Taxa de Erro (desvio-padrão) [intervalo de confiança empírico].

Os resultados foram satisfatórios para este problema onde as classes são bem separadas e de uma maneira geral todos os modelos apresentaram um bom desempenho. Note que para todos os modelos houve decrescimento das médias das TE's com o aumento do conjunto de treinamento. Para $n_{tre} = 200$ o melhor desempenho foi do ajuste com MFN e MFtA $(0,88\%)$, com $n_{tre} = 300$ a menos TE foi obtida com os ajustes com MFt e MFtA $(0,84\%)$. A menor média para TE obtida foi para o modelo de MFtA, com $n_{tre} = 500$ e $n_{tre} = 1000$, para este último tamanho de conjunto de treinamento o modelo de MFtA apresentou o mesmo desempenho médio $(0,81\%)$. O modelo que teve o pior desempenho foi a MFNA, tendo a menor média para TE igual a $0,84\%$.

Neste problema, não empregamos os modelos de misturas para gerar as classes, mas apenas geramos cada uma com distribuição diferente, a Classe 1 gerada por uma distribuição t assimétrica e a Classe 2 gerada por uma distribuição normal assimétrica. Apesar dessa característica, de maneira geral, todos os modelos apresentaram desempenho similar, selecionando apenas uma componente para ambas as classes, como adequado. Apesar destas diferenças em termos da média das taxas de erro, observamos que os IC estão todos sobrepostos, sugerindo que os ajustes tiveram um comportamento equivalente. Estes resultados são, provavelmente, devido ao fato de que as classes estão relativamente separadas

Na Figura **??**, temos os *boxplots* para as TE's obtidas neste problema, onde observamos os comportamentos equivalentes dos modelos em relação às suas TE's.

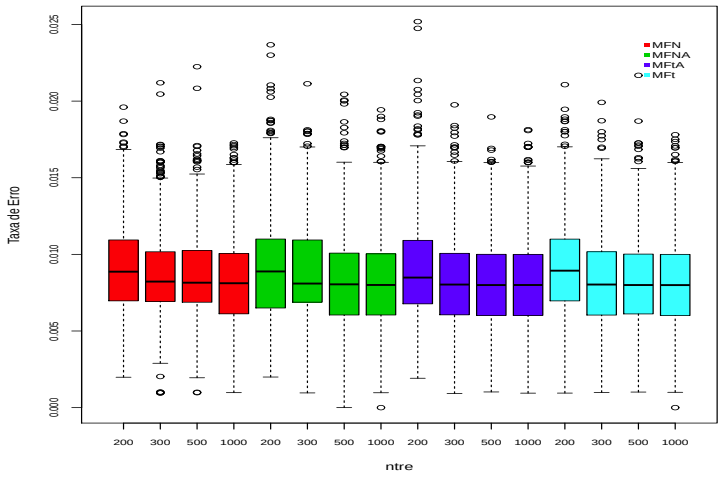


Figura 5.9: Taxas de erro do Problema 7.

5.4.4 **Problema** 8

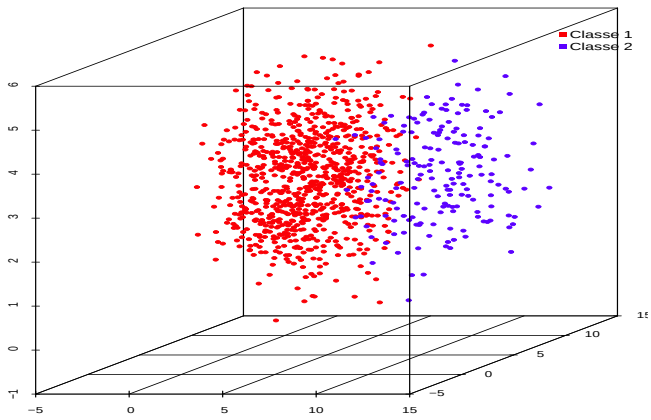


Figura 5.10: Gráfico do Problema 8.

No Problema 8, simulamos as observações com os mesmo parâmetros do Problema 4, onde a Classe 1 foi gerada com probabilidade $P_1 = 0,8$ e a Classe 2 com probabilidade $P_2 = 0,2$, mas estendemos a dimensão dos dados para $p = 3$, e sua estrutura pode ser observada na Figura **??**.

A partir da configuração bidimensional, foi acrescentada uma variável independente e redundante para efeito de discriminação, obtendo-se portanto um vetor de observações de dimensão $p = 3$. A variável independente e redundante gerada tem distribuição Normal$(3,1)$. Na Figura **??** podemos observar o comportamento das variáveis desse problema.

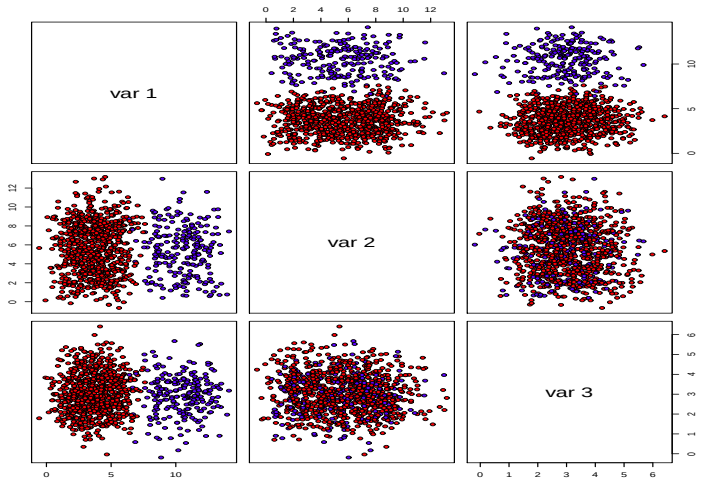


Figura 5.11: Gráfico de dispersão do Problema 8, com $p = 3$.

Na Tabela **??** são apresentadas as TE's de classificação do conjunto de teste, segundo o tamanho do conjunto de treinamento, para cada uma das distribuições.

n_{tre}	MFN	MFNA	MFt	MFtA
200	0,50(0,30) [0,10;1,12]	0,55(0,38) [0,10;1,41]	0,52(0,28) [0,10;1,18]	0,53(0,30) [0,10;1,20]
300	0,38(0,20) [0,09;0,80]	0,47(0,25) [0,10;1,01]	0,43(0,24) [0,00;0,96]	0,46(0,25) [0,10;1,06]
500	0,37(0,20) [0,00;0,82]	0,41(0,21) [0,10;0,90]	0,38(0,22) [0,00;0,82]	0,41(0,22) [0,10;0,90]
1000	**0,30**(0,18) [0,00;0,70]	0,32(0,19) [0,00;0,71]	**0,30**(0,18) [0,00;0,70]	0,33(0,19) [0,00;0,80]

Tabela 5.5: Problema 8 - Média da Taxa de Erro (desvio-padrão) [intervalo de confiança empírico].

Mais uma vez observamos que para todos os modelos houve decrescimento das médias das taxas com o aumento do conjunto de treinamento. Para os conjuntos de treinamento de tamanhos 200, 300 e 500, as menores médias para TE foram obtidas para MFN (0,50%, ,38% e 0,37%, respectivamente). Com $n_{tre} = 1000$ a menor média para TE foi obtida para MFN e MFt $(0,30\%)$. Mas, observe que os IC's estão sobrepostos e, como esperado, o menor desvio padrão é observados para grandes amostras.

Na Figura **??** temos os *boxplots* das TE's obtidas neste problema. Observe nos *boxplots*, o comportamento equivalente entre os modelos e a maior variabilidade com $n_{tre} = 200$, como discutido anteriormente.

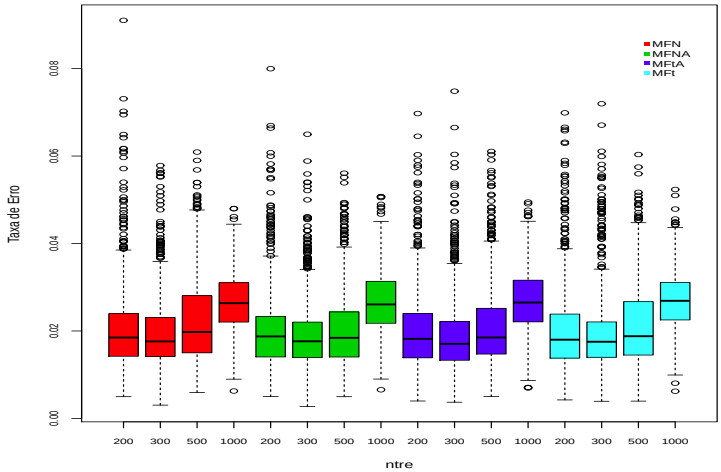


Figura 5.12: Taxas de erro do Problema 8.

A seleção do número de componentes foi similar ao apresentado no Problema 4, todos os modelos selecionaram duas componentes para a Classe 1. E para a Classe 2, foi selecionada uma componente, com exceção para o conjunto de treinamento de tamanho 1000, onde para todos os modelos foram selecionadas duas componentes.

Portanto, o acréscimo da dimensão redundante não afetou de maneira significativa os modelos.

5.5 Aplicação com dados reais

Nesta seção empregaremos a metodologia proposta neste trabalho considerando três conjuntos de dados reais. Inicialmente, abordaremos dois conjuntos de dados já analisados na literatura por diversos classificadores, com abordagens estatística e não estatística, com o objetivo de avaliar o classificador.

Como não dispomos das probabilidades *a priori* das classes, empregamos nestes conjuntos de dados o classificador de máxima verossimilhança, ou seja, consideramos as probabilidades *a priori* de cada classe como sendo iguais, isto é, não informativa.

E além de separar as observações em conjunto de *treinamento e teste*, empregamos o procedimento conhecido como *k-fold*, por ser mais confiável para estimar a TE, pois todas as observações do conjunto de dados são classificadas (para detalhes, ver Hastie *et al.* [10, Seção. 7.10].

5.5.1 Dados 1: - *Caranguejos Leptograpsus*.

O primeiro conjunto de dados que iremos descrever pode ser encontrado no pacote `MASS` do *software R*, denominado `crabs`, o *Caranguejos Leptograpsus*. Trata-se de um estudo sobre uma espécie de caranguejo, do gênero *Leptograpsus*, que é classificada segundo a sua cor, laranja ou azul. Mas, com a perda da cor, esperava-se que as diferenças morfológicas permitiriam classificá-los. O conjunto de dados tem um total de 200 observações, 100 de cada sexo e de cada espécie, coletadas em Fremantle, Austrália Ocidental. Para cada observação temos as medidas da largura frontal do casco (FL), largura posterior do casco (RW), comprimento central do casco (CL), comprimento máximo do casco (CW) e altura do corpo (BD), em milímetros (Ripley [24, Seção. 1.4]). Na Figura **??**, apresentamos o histograma de cada variável em estudo, segundo a sua classificação, onde a Classe 1 é a espécie *azul* e a Classe 2 é a espécie *laranja* . Note que de uma maneira geral, as variáveis apresentam assimetria e multimodalidade, características intrínsecas aos modelos de misturas finitas de distribuições pertencentes à família NAI.

Capítulo 6

Conclusões e Sugestões

Neste capítulo, discutimos os principais resultados obtidos com a proposta deste trabalho para os problemas simulados e as aplicações com dados reais.

6.1 Discussões e Conclusões

Neste trabalho apresentamos a Análise Discriminante, que é uma ferramenta amplamente utilizada, mas com uma abordagem diferente. Sugerimos o emprego de distribuições de probabilidades provenientes da Família Normal Assimétrica Independente para o desenvolvimento de um procedimento de classificação.

Observamos no geral, em cada simulação realizada, que o classificador proposto conseguiu se ajustar aos diferentes problemas estudados, desde os mais simples aos mais complexos em termos de modelagem e separação das classes.

Além de TE's razoáveis, ainda temos informações sobre a probabilidade de um objeto pertencer a uma determinada classe (probabilidade *a posteriori*), uma vez que adaptamos o classificador de Bayes, que consiste em calcular essas probabilidades e alocar cada objeto na classe em que foi obtida a maior probabilidade *a posteriori*.

Em alguns problemas, os modelos considerados se ajustaram de tal maneira que o número de componentes necessário para a estimação das densidades foi menor quando comparado com o modelo empregado para a simulação, diminuindo assim tempo e quantidade de parâmetros estimados, sem perda de informação.

O desempenho do classificador com os dados reais também foi satisfatório. Apenas verificamos a dificuldade do classificador, da forma como propomos variando o número de componentes para as misturas, incorporar um número maior de variáveis. É possível que isso decorra devido ao fato de que os agrupamentos iniciais dados pelo *k-means*, formam componentes (grupos) com poucas observações, o que gera estimativas singulares para as matrizes de dispersão nas componentes da mistura.

Em particular, com os dados reais *Caranguejos Leptograpsus*, o classificador não encontrou dificuldades, pois sua fronteira de decisão é estritamente linear sendo, portanto, um conjunto bem separado. Como mencionado anteriormente, o classificador proposto conseguiu se adaptar a um problema simples como este, classificando as observações corretamente como os classificadores usuais.

Realmente, os modelos de MF com densidades provenientes da família NAI conseguem modelar situações em que os modelo usuais não se adequariam, seja por um problema de multimodalidade ou assimetria. Mas, vale ressaltar que em situações simples não tivemos perda em empregar esses modelos, pois eles se adaptaram satisfatoriamente.

Como mencionado, com os dados reais, foi verificada a dificuldade da modelagem quando dispomos de poucas observações por classe relativas à dimensão do vetor de características. O principal problema observado foi a questão da estimação das matrizes de escala, que resultam em matrizes singulares. Uma sugestão seria impor a restrição de que as matrizes de escala sejam todas diagonais nas componentes da mistura, o que resultaria em um classificador "ingênuo"de Bayes (denominado na literatura de Naive Bayes, ver por, exemplo Hastie *et al.* [10, Seção. 6.6]) com as distribuições condicionais aproximadas por misturas finitas de densidades.

Apêndice A

Programas

A.1 Simulações

```
#-----------------------------------# SIMULAÇÕES #-----------------------------
library(mvtnorm)
library(mixsmsn)
library(MASS)
library(sn)
p = 2   #dimensão das observações
#-------------------------------------# PROBLEMA 1 #---------------------------
pi1       <- 0.6    #probabilidade a priori da classe 1
pi2       <- 1-pi1  #probabilidade a priori da classe 2
ntre      <- 200    #tamanho do conjunto de treinamento 200, 300, 500 e 1000
#classe 1- o modelo foi uma mistura de três normais
mu11    <- c(5,5)
sigma11 <- matrix(c(1,0,0,3),2,2)
shape11 <- c(0,0)
nu11    <- NULL
mu12    <- c(6,11)
sigma12 <- matrix(c(5,0,0,5),2,2)
shape12 <- c(0,0)
nu12    <- NULL
mu13    <- c(11,11)
sigma13 <- matrix(c(3,0,0,1),2,2)
shape13 <- c(0,0)
```

```
nu13    <- NULL
#classe 2- o modelo foi uma mistura de duas normais
mu21    <- c(10,6)
sigma21 <- matrix(c(1,0,0,1),2,2)
shape21 <- c(0,0)
nu21    <- NULL
mu22    <- c(6,16)
sigma22 <- matrix(c(1,0,0,1),2,2)
shape22 <- c(0,0)
nu22    <- NULL
c <- c(0,0)
n <- 1
pii1 <- c(0.4,0.2,0.4)
pii2 <- c(0.5,0.5)
arg11 <- list(mu=mu11,Sigma=sigma11,shape=shape11,nu=nu11)
arg12 <- list(mu=mu12,Sigma=sigma12,shape=shape12,nu=nu12)
arg13 <- list(mu=mu13,Sigma=sigma13,shape=shape13,nu=nu13)
arg21 <- list(mu=mu21,Sigma=sigma21,shape=shape21,nu=nu21)
arg22 <- list(mu=mu22,Sigma=sigma22,shape=shape22,nu=nu22)
for(k in 1: 1000){
#-------------------- gerando as misturas (para as classes 1 e 2) --------------
l <- 0
c = q = vector()  #vetor ce classificação para as observações de treinamento
x <- matrix(0,ntre,2)  #matriz de observações de treinamento
while (l <= ntre) {
for(i in 1:ntre){
 q[i]=runif(1)
 if(q[i] < pi1){
 x[i,] <- rmmix(1,p=pii1,"Normal", list(arg11,arg12,arg13))
 c[i]  <- 1
 l     <- l+1
  }
  else {
x[i,] <- rmmix(1,p=pii2,"Normal",list(arg21,arg22))
c[i]  <- 2
}}}
#-------------------------------------# PROBLEMA 2 #----------------------------
pi1  <- 0.5    #probabilidade a priori da classe 1
pi2  <- 1-pi1   #probabilidade a priori da classe 2
```

```
ntre <- 200    #tamanho do conjunto de treinamento 200, 300, 500 e 1000
#classe 1, o modelo foi uma mistura de cinco normais
mu11=c(10,10)
sigma11=matrix(c(3,0,0,1),2,2)
shape11=c(0,0)
nu11=NULL
mu12=c(5,11)
sigma12=matrix(c(1,-0.5,-0.5,1),2,2)
shape12=c(0,0)
nu12=NULL
mu13=c(3,16)
sigma13=matrix(c(1,0,0,3),2,2)
shape13=c(0,0)
nu13=NULL
mu14=c(5,21)
sigma14=matrix(c(1,0.5,0.5,1),2,2)
shape14=c(0,0)
nu14=NULL
mu15=c(10,22)
sigma15=matrix(c(3,0,0,1),2,2)
shape15=c(0,0)
nu15=NULL
#classe 2, o modelo foi uma mistura de cinco normais
mu21=c(11,4)
sigma21=matrix(c(3,0,0,1),2,2)
shape21=c(0,0)
nu21=NULL
mu22=c(16,5)
sigma22=matrix(c(1,0.5,0.5,1),2,2)
shape22=c(0,0)
nu22=NULL
mu23=c(18,10)
sigma23=matrix(c(1,0,0,3),2,2)
shape23=c(0,0)
nu23=NULL
mu24=c(16,15)
sigma24=matrix(c(1,-0.5,-0.5,1),2,2)
shape24=c(0,0)
nu24=NULL
```

```
mu25=c(11,16)
sigma25=matrix(c(3,0,0,1),2,2)
shape25=c(0,0)
nu25=NULL
c=c(0,0)
n=1
pii1=c(0.2,0.2,0.2,0.2,0.2)
pii2=c(0.2,0.2,0.2,0.2,0.2)
arg11=list(mu=mu11,Sigma=sigma11,shape=shape11,nu=nu11)
arg12=list(mu=mu12,Sigma=sigma12,shape=shape12,nu=nu12)
arg13=list(mu=mu13,Sigma=sigma13,shape=shape13,nu=nu13)
arg14=list(mu=mu14,Sigma=sigma14,shape=shape14,nu=nu14)
arg15=list(mu=mu15,Sigma=sigma15,shape=shape15,nu=nu15)
arg21=list(mu=mu21,Sigma=sigma21,shape=shape21,nu=nu21)
arg22=list(mu=mu22,Sigma=sigma22,shape=shape22,nu=nu22)
arg23=list(mu=mu23,Sigma=sigma23,shape=shape23,nu=nu23)
arg24=list(mu=mu24,Sigma=sigma24,shape=shape24,nu=nu24)
arg25=list(mu=mu25,Sigma=sigma25,shape=shape25,nu=nu25)
#----------------- gerando as misturas (para as classes 1 e 2) -----------------
l=0
c=q=vector()
x=matrix(0,ntre,2)
while (l <= ntre) {
for(i in 1:ntre){
q[i]=runif(1)
if(q[i] < pi1){
x[i,]=rmmix(1,p=pii1,"Normal", list(arg11,arg12,arg13,arg14,arg15))
c[i]=1
l=l+1
} else {

x[i,]=rmmix(1,p=pii2,"Normal",list(arg21,arg22,arg23,arg24,arg25))
c[i]=2
}
}
}
#-------------------------------------# PROBLEMA 3 #----------------------------
pi1 = 0.4    #probabilidade a priori da classe 1
pi2 = 1-pi1   #probabilidade a priori da classe 2
```

```
ntre = 200   #tamanho do conjunto de treinamento 200, 300, 500 e 1000.
#classe 1 - o modelo foi uma normal
mu11= c(rep(0,d))
sigma11=diag(rep(1,d))
shape11=c(0,0)
nu11=NULL
#classe 2 - o modelo foi uma normal
mu21=c(rep(0,d))
sigma21=8*diag(rep(1,d))
shape21=c(0,0)
nu21=NULL
c=c(0,0)
n=1
pii1=1
pii2=1
arg11=list(mu=mu11,Sigma=sigma11,shape=shape11,nu=nu11)
arg21=list(mu=mu21,Sigma=sigma21,shape=shape21,nu=nu21)
#------------------------ gerando as misturas (para as classes 1 e 2) ----------
l=0
c=q=vector()
x=matrix(0,ntre,2)
while (l <= ntre) {
for(i in 1:ntre){
q[i]=runif(1)
if(q[i] < pi1){
x[i,]=rmmix(1,p=pii1,"Normal", list(arg11))
c[i]=1
l=l+1
} else {
x[i,]=rmmix(1,p=pii2,"Normal",list(arg21))
c[i]=2
}}}
#-------------------------------------# PROBLEMA 4 #----------------------------
pi1 = 0.8    #probabilidade a priori da classe 1
pi2 = 1-pi1   #probabilidade a priori da classe 2
ntre = 200   #tamanho do conjunto de treinamento 200, 300, 500 e 1000
#classe 1 - o modelo foi uma mistura de cinco normais
mu11=c(5,2)
sigma11=matrix(c(1,-0.5,-0.5,1),2,2)
```

```
shape11=c(0,0)
nu11=NULL
mu12=c(3,3)
sigma12=diag(rep(1,d))
shape12=c(0,0)
nu12=NULL
mu13=c(2,5)
sigma13=diag(rep(1,d))
shape13=c(0,0)
nu13=NULL
mu14=c(3,7)
sigma14=diag(rep(1,d))
shape14=c(0,0)
nu14=NULL
mu15=c(5,8)
sigma15=diag(rep(1,d))
shape15=c(0,0)
nu15=NULL
#classe 2 - o modelo foi uma mistura de cinco normais
mu21=c(9,5)
sigma21=diag(rep(1,d))
shape21=c(0,0)
nu21=NULL
mu22=c(11,6)
sigma22=diag(rep(1,d))
shape22=c(0,0)
nu22=NULL
mu23=c(12,8)
sigma23=diag(rep(1,d))
shape23=c(0,0)
nu23=NULL
mu24=c(11,10)
sigma24=diag(rep(1,d))
shape24=c(0,0)
nu24=NULL
mu25=c(9,11)
sigma25=diag(rep(1,d))
shape25=c(0,0)
nu25=NULL
```

```
c=c(0,0)
n=1
pii1=c(0.2,0.2,0.2,0.2,0.2)
pii2=c(0.2,0.2,0.2,0.2,0.2)
arg11=list(mu=mu11,Sigma=sigma11,shape=shape11,nu=nu11)
arg12=list(mu=mu12,Sigma=sigma12,shape=shape12,nu=nu12)
arg13=list(mu=mu13,Sigma=sigma13,shape=shape13,nu=nu13)
arg14=list(mu=mu14,Sigma=sigma14,shape=shape14,nu=nu14)
arg15=list(mu=mu15,Sigma=sigma15,shape=shape15,nu=nu15)
arg21=list(mu=mu21,Sigma=sigma21,shape=shape21,nu=nu21)
arg22=list(mu=mu22,Sigma=sigma22,shape=shape22,nu=nu22)
arg23=list(mu=mu23,Sigma=sigma23,shape=shape23,nu=nu23)
arg24=list(mu=mu24,Sigma=sigma24,shape=shape24,nu=nu24)
arg25=list(mu=mu25,Sigma=sigma25,shape=shape25,nu=nu25)
#---------------------- gerando as misturas (para as classes 1 e 2) -----------
l=0
c=q=vector()
x=matrix(0,ntre,2)
while (l <= ntre) {
for(i in 1:ntre){
q[i]=runif(1)
if(q[i] < pi1){
x[i,]=rmmix(1,p=pii1,"Normal", list(arg11,arg12,arg13,arg14,arg15))
c[i]=1
l=l+1
} else {
x[i,]=rmmix(1,p=pii2,"Normal",list(arg21,arg22,arg23,arg24,arg25))
c[i]=2
}}}
#-------------------------------------# PROBLEMA 5 #----------------------------
pi1 = 0.4    #probabilidade a priori da classe 1
pi2 = 1-pi1   #probabilidade a priori da classe 2
ntre = 200   #tamanho do conjunto de treinamento 200, , 300, 500 e 1000
#classe 1 - o modelo foi uma normal
mu11= c(rep(40,d))
sigma11=matrix(c(180,0,0,180),d,d,byrow=F)
shape11=c(0,0)
nu11=NULL
#classe 2 - o modelo foi uma normal
```

```
mu21=c(rep(0,d))
sigma21=matrix(c(100,50,50,100),d,d,byrow=F)
shape21=c(0,0)
nu21=NULL
c=c(0,0)
n=1
pii1=1
pii2=1
arg11=list(mu=mu11,Sigma=sigma11,shape=shape11,nu=nu11)
arg21=list(mu=mu21,Sigma=sigma21,shape=shape21,nu=nu21)
#---------------------- gerando as misturas (para as classes 1 e 2) -----------
l=0
c=q=vector()
x=matrix(0,ntre,2)
while (l <= ntre) {
for(i in 1:ntre){
q[i]=runif(1)
if(q[i] < pi1){
x[i,]=rmmix(1,p=pii1,"Normal", list(arg11))
c[i]=1
l=l+1
} else {

x[i,]=rmmix(1,p=pii2,"Normal",list(arg21))
c[i]=2
}}}
#-----------------------------------# PROBLEMA 6 #----------------------------
pi1 = 0.6    #probabilidade a priori da classe 1
pi2 = 1-pi1   #probabilidade a priori da classe 2
ntre = 200   #tamanho do conjunto de treinamento 200, 300, 500 e 1000
#classe 1 - o modelo foi uma mistura de três t assimétricas
mu11=c(0,6)
sigma11=matrix(c(8,10.74,10.74,30),2,2)
shape11=c(0,0)
nu11=10
mu12=c(15,15)
sigma12=matrix(c(40,5,30,2),2,2)
shape12=c(0,0)
nu12=10
```

```
mu13=c(30,8)
sigma13=matrix(c(15,-10,-10,10),2,2)
shape13=c(0,0)
nu13=10
#classe 2 - o modelo foi uma mistura de duas t assimétricas
mu21=c(6,3.5)
sigma21=matrix(c(10,-5,-5,10),2,2)
shape21=c(0,0)
nu21=10
mu22=c(20,3.5)
sigma22=matrix(c(10,5,5,10),2,2)
shape22=c(0,0)
nu22=10
c=c(0,0)
n=1
pii1=c(0.3,0.3,0.4)
pii2=c(0.5,0.5)
arg11=list(mu=mu11,Sigma=sigma11,shape=shape11,nu=nu11)
arg12=list(mu=mu12,Sigma=sigma12,shape=shape12,nu=nu12)
arg13=list(mu=mu13,Sigma=sigma13,shape=shape13,nu=nu13)
arg21=list(mu=mu21,Sigma=sigma21,shape=shape21,nu=nu21)
arg22=list(mu=mu22,Sigma=sigma22,shape=shape22,nu=nu22)
#----------------------- gerando as misturas (para as classes 1 e 2) -----------
l=0
c=q=vector()
x=matrix(0,ntre,2)
while (l <= ntre) {
for(i in 1:ntre){
q[i]=runif(1)
if(q[i] < pi1){
x[i,]=rmmix(1,p=pii1,"Skew.t", list(arg11,arg12,arg13))
c[i]=1
l=l+1
} else {
x[i,]=rmmix(1,p=pii2,"Skew.t",list(arg21,arg22))
c[i]=2
}}}
#-------------------------------------# PROBLEMA 7 #----------------------------
pi1 = 0.4    #probabilidade a priori da classe 1
```

```
pi2 = 1-pi1   #probabilidade a priori da classe 2
ntre = 200   #tamanho do conjunto de treinamento 200, , 300, 500 e 1000
#classe 1 - o modelo foi uma t assimétrica
mu11= c(rep(40,d))
sigma11=matrix(c(60,20,20,60),d,d,byrow=F)
shape11=c(-2,-2)
nu11=10
#classe 2 - o modelo foi uma normal assimétricas
mu21=c(rep(0,d))
sigma21=matrix(c(80,30,30,80),d,d,byrow=F)
shape21=c(1,1)
nu21=NULL
c=c(0,0)
n=1
pii1=1
pii2=1
arg11=list(mu=mu11,Sigma=sigma11,shape=shape11,nu=nu11)
arg21=list(mu=mu21,Sigma=sigma21,shape=shape21,nu=nu21)
for(k in 1: 1000){
#------------------------- gerando as misturas (para as classes 1 e 2) ---------
l=0
c=q=vector()
x=matrix(0,ntre,2)
while (l <= n1) {
for(i in 1:ntre){
q[i]=runif(1)
if(q[i] < pi1){
x[i,]=rmmix(1,p=pii1,"Skew.t", list(arg11))
c[i]=1
l=l+1
} else {

x[i,]=rmmix(1,p=pii2,"Skew.normal",list(arg21))
c[i]=2
}}}
#-------------------------------------# PROBLEMA 7 #----------------------------
pi1 = 0.8    #probabilidade a priori da classe 1
pi2 = 1-pi1   #probabilidade a priori da classe 2
ntre = 200   #tamanho do conjunto de treinamento 200, 300, 500 e 1000
```

```
#classe 1, o modelo foi uma mistura de cinco normais
mu11=c(5,2)
sigma11=matrix(c(1,-0.5,-0.5,1),2,2)
shape11=c(0,0)
nu11=NULL
mu12=c(3,3)
sigma12=diag(rep(1,d))
shape12=c(0,0)
nu12=NULL
mu13=c(2,5)
sigma13=diag(rep(1,d))
shape13=c(0,0)
nu13=NULL
mu14=c(3,7)
sigma14=diag(rep(1,d))
shape14=c(0,0)
nu14=NULL
mu15=c(5,8)
sigma15=diag(rep(1,d))
shape15=c(0,0)
nu15=NULL
#classe 2, o modelo foi uma mistura de cinco normais
mu21=c(9,5)
sigma21=diag(rep(1,d))
shape21=c(0,0)
nu21=NULL
mu22=c(11,6)
sigma22=diag(rep(1,d))
shape22=c(0,0)
nu22=NULL
mu23=c(12,8)
sigma23=diag(rep(1,d))
shape23=c(0,0)
nu23=NULL
mu24=c(11,10)
sigma24=diag(rep(1,d))
shape24=c(0,0)
nu24=NULL
mu25=c(9,11)
```

```
sigma25=diag(rep(1,d))
shape25=c(0,0)
nu25=NULL
c=c(0,0)
n=1
pii1=c(0.2,0.2,0.2,0.2,0.2)
pii2=c(0.2,0.2,0.2,0.2,0.2)
arg11=list(mu=mu11,Sigma=sigma11,shape=shape11,nu=nu11)
arg12=list(mu=mu12,Sigma=sigma12,shape=shape12,nu=nu12)
arg13=list(mu=mu13,Sigma=sigma13,shape=shape13,nu=nu13)
arg14=list(mu=mu14,Sigma=sigma14,shape=shape14,nu=nu14)
arg15=list(mu=mu15,Sigma=sigma15,shape=shape15,nu=nu15)
arg21=list(mu=mu21,Sigma=sigma21,shape=shape21,nu=nu21)
arg22=list(mu=mu22,Sigma=sigma22,shape=shape22,nu=nu22)
arg23=list(mu=mu23,Sigma=sigma23,shape=shape23,nu=nu23)
arg24=list(mu=mu24,Sigma=sigma24,shape=shape24,nu=nu24)
arg25=list(mu=mu25,Sigma=sigma25,shape=shape25,nu=nu25)
#-------------------------- gerando as misturas (para as classes 1 e 2) --------
l=0
c=q=vector()
x=matrix(0,ntre,2)
while (l <= n1) {
for(i in 1:ntre){
q[i]=runif(1)
if(q[i] < pi1){
x[i,]=rmmix(1,p=pii1,"Normal", list(arg11,arg12,arg13,arg14,arg15))
c[i]=1
l=l+1
} else {

x[i,]=rmmix(1,p=pii2,"Normal",list(arg21,arg22,arg23,arg24,arg25))
c[i]=2
}}}
#------------------------------ matriz de observações -----------------------
amostra_ini=data.frame(x,c)
amostra_ini$c=as.factor(amostra_ini$c)
#------------- gerando observações para classificação --------------------------
ntes=1000                  #observações a serem classificadas
c2=q2=vector()
```

```
y=matrix(0,ntes,2)
for(i in 1:ntes){
q2[i]=runif(1)
if(q2[i] < pi1){
y[i,]=rmmix(1,p=pii1,"Normal", list(arg11,arg12,arg13,arg14,arg15))
c2[i]=1
} else {
y[i,]=rmmix(1,p=pii2,"Normal",list(arg21,arg22,arg23,arg24,arg25))
c2[i]=2
}}
#------------------------------ matriz de observações -------------------------
amostraobs_ini=data.frame(y,c2)
amostraobs_ini$c2=as.factor(amostraobs_ini$c2)
#----------------------------------- Gerando a terceira dimensão --------------
X3=matrix(c(rnorm(ntre, mean = 3, sd = 1)))
X3obs=matrix(c(rnorm(1000, mean = 3, sd = 1)))
x=cbind(amostra_ini$X1,amostra_ini$X2,X3)
y=cbind(amostraobs_ini$X1,amostraobs_ini$X2,X3obs)
#----------------------- matriz de observações para o PROBLEMA 8 ---------------
amostra=cbind(amostra_ini$X1,amostra_ini$X2,X3,amostra_ini$c)
amostraobs=cbind(amostraobs_ini$X1,amostraobs_ini$X2,X3obs,amostraobs_ini$c2)
#------------------ matriz de observações para os demais problemas -------------
amostra   <- data.frame(x,c)  ### conjunto de treinamento
amostra$c <- as.factor(amostra$c) #para garantir que a coluna c é de fatores
#------------------------------ separando classes -----------------------------
g1 <- amostra[c==1,] ##conjunto de treinamento classe 1
g2 <- amostra[c==2,] ##conjunto de treinamento classe 2
#----------- gerando observações para classificação ---------------------------
ntes <- 1000      #observações a serem classificadas
c2   <- vector() #vetor que irá guardar a classificação
y    <- matrix(0,ntes,2) #ntes é o tamanho da amostra teste
for(i in 1:ntes){
  q2=runif(1)
  if(q2 < pi1){
       y[i,]<- rmmix(1,p=pii1,"Normal", list(arg11,arg12,arg13))
   c2[i]<- 1
    }
    else {
   y[i,] <- rmmix(1,p=pii2,"Normal",list(arg21,arg22))
```

```
   c2[i] <- 2
}}
#-------------------------matriz de observações para teste --------------------
amostraobs    <- data.frame(y,c2)
amostraobs$c2 <- as.factor(amostraobs$c2)
head(amostraobs)
#----------------------   separando classes das observações de teste -----------
g1obs<-amostraobs[c2==1,]
g2obs<-amostraobs[c2==2,]
mydir <- " "
#AJUSTANDO PELA NORMAL
iter2_N=iter1_N=ncomp1_N = ncomp2_N = vector()
#ajustando as famílias e o número de componentes indicado, para a classe 1
w1_N<- smsn.search(y=g1[,1:2],nu=2, g.min = 1, g.max = 5, family = "Normal",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
ncomp1_N<- which.min(w1_N$criteria)
iter1_N<- w1_N$best.model$iter
#ajustando as famílias e o número de componentes indicado, para a classe 2
w2_N<-smsn.search(g2[,1:2],nu=2, g.min = 1, g.max = 5, family = "Normal",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
ncomp2_N   <- which.min(w2_N$criteria)
iter2_N<-w2_N$best.model$iter
#------------------- INSERINDO ESTIMATIVAS EM OBJETOS --------------------------
p1_N<-matrix(c( w1_N$best.model$pii),ncomp1_N,1)
p2_N<-matrix(c( w2_N$best.model$pii),ncomp2_N,1)
#alocando as estimativas dos parâmetros da classe 1 nas matrizes
mu_est1_N     <-w1_N$best.model$mu
sigma_est1_N <-w1_N$best.model$Sigma
#alocando as estimativas dos parâmetros da classe 2 nas matrizes
mu_est2_N     <-w2_N$best.model$mu
sigma_est2_N <-w2_N$best.model$Sigma
#--------------------- DENSIDADES PARA CLASSIFICAÇÃO --------------------------
probpost1_N = probpost2_N=num1_N=num2_N=vector()
sf1_N = pf1_N = f1_N = matrix(rep(0,ncomp1_N),ntes,ncomp1_N)
sf2_N = pf2_N = f2_N = matrix(rep(0,ncomp2_N),ntes,ncomp2_N)
##CLASSE 1
for(j in 1:ncomp1_N){
for(i in 1:nrow(y)){
f1_N[i,j]<-dmvnorm(y[i,1:2], mean=mu_est1_N[[j]],
```

```
    sigma=sigma_est1_N[[j]]%*%sigma_est1_N[[j]])
pf1_N[i,j]<-p1_N[j]*f1_N[i,j]
}}
sf1_N<-apply(pf1_N, 1, sum)
num1_N<-sf1_N*(nrow(g1)/ntre)
#CLASSE 2
for(j in 1:ncomp2_N){
for(i in 1:nrow(y)){
f2_N[i,j]<-dmvnorm(y[i,1:2], mean=mu_est2_N[[j]],
        sigma=sigma_est2_N[[j]]%*%sigma_est2_N[[j]])
pf2_N[i,j]<-p2_N[j]*f2_N[i,j]
}}
sf2_N<-apply(pf2_N, 1, sum)
num2_N<-sf2_N*(nrow(g2)/ntre)
#------------------------ PROBABILIDADES POSTERIORIS PARA CADA CLASSE ----------
for(i in 1:nrow(y)){
probpost1_N[i]<-num1_N[i]/(num1_N[i]+num2_N[i])
probpost2_N[i]<-num2_N[i]/(num1_N[i]+num2_N[i])
}
#----------------------------- comparando densidades --------------------------
class_N<-vector()
for(i in 1:nrow(y)){
if(num1_N[i] > num2_N[i]){

class_N[i]<-1
} else {
class_N[i]<-2
}}
mcomp_N<-matrix(c(probpost1_N,probpost2_N,class_N,c2),ntes,4)
n1=table(c2)[1]       #número de observações da classe 1
n2=table(c2)[2]     #número de observações da classe 2
bic1_N=w1_N$best.model$bic
bic2_N=w2_N$best.model$bic
conf_N      <- matrix(c(table(class_N,c2)),2,2)   #tabela de confundimento
n12_g1_N <- conf_N[1,2]   #n°. de observações da classe 1 alocada na classe 2
n21_g2_N <- conf_N[2,1] #n°. de observações da classe 2 alocada na classe 1
erro_total1_N <- n12_g1_N/n1 #erro total de classificação da classe 1 em 2
erro_total2_N <- n21_g2_N/n2 #erro total de classificação da classe 2 em 1
N_g1<-cbind(k=k ,n1comp=ncomp1_N, bic=bic1_N,erro=erro_total1_N )
```

```
N_g2<-cbind(k=k,n2comp=ncomp2_N, bic=bic2_N, erro=erro_total2_N)
write.table(N_g1,paste(mydir,"N_g1.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
write.table(N_g2,paste(mydir,"N_g2.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
mydir <- " "
#AJUSTANDO PELA SKEW-NORMAL
ncomp_SN<-matrix(c(0,0,0),4,3)
iter2_SN=iter1_SN=ncomp1_SN = ncomp2_SN = vector()
w1_SN <- smsn.search(g1[,1:2],nu=2, g.min = 1, g.max = 5, family = "Skew.normal",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
ncomp1_SN<- which.min(w1_SN$criteria)
iter1_SN<-w1_SN$best.model$iter
#ajustando as famílias e o número de componentes indicado, para a classe 2
w2_SN <- smsn.search(g2[,1:2],nu=2, g.min = 1, g.max = 5, family = "Skew.normal",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
ncomp2_SN <- which.min(w2_SN$criteria)
iter2_SN<-w2_SN$best.model$iter
p1_SN<-matrix(c( w1_SN$best.model$pii),ncomp1_SN,1)
p2_SN<-matrix(c( w2_SN$best.model$pii),ncomp2_SN,1)
mu_est1_SN<-w1_SN$best.model$mu
sigma_est1_SN<-w1_SN$best.model$Sigma
shape_est1_SN<-w1_SN$best.model$shape
#alocando as estimativas dos parâmetros da classe 2 nas matrizes
mu_est2_SN<-w2_SN$best.model$mu
sigma_est2_SN<-w2_SN$best.model$Sigma
shape_est2_SN<-w2_SN$best.model$shape
library(sn)
probpost1_SN=probpost2_SN=num1_SN=num2_SN=vector()
sf1_SN = pf1_SN = f1_SN = matrix(rep(0,ncomp1_SN),ntes,ncomp1_SN)
sf2_SN = pf2_SN = f2_SN = matrix(rep(0,ncomp2_SN),ntes,ncomp2_SN)
#CLASSE 1
for(j in 1:ncomp1_SN){
for(i in 1:nrow(y)){
f1_SN[i,j]<-dmsn(y[i,1:2], xi = mu_est1_SN[[j]],
    Omega=sigma_est1_SN[[j]]%*%sigma_est1_SN[[j]], alpha=shape_est1_SN[[j]])
pf1_SN[i,j]<-p1_SN[j]*f1_SN[i,j]
}}
sf1_SN<-apply(pf1_SN, 1, sum)
```

```
num1_SN<-sf1_SN*(nrow(g1)/ntre)
##CLASSE 2
for(j in 1:ncomp2_SN){
for(i in 1:nrow(y)){
 f2_SN[i,j]<-dmsn(y[i,1:2], xi = mu_est2_SN[[j]],
     Omega=sigma_est2_SN[[j]]%*%sigma_est2_SN[[j]], alpha=shape_est2_SN[[j]])
 pf2_SN[i,j]<-p2_SN[j]*f2_SN[i,j]
}}
sf2_SN<-apply(pf2_SN, 1, sum)
num2_SN<-sf2_SN*(nrow(g2)/ntre)
for(i in 1:nrow(y)){
probpost1_SN[i]<-num1_SN[i]/(num1_SN[i]+num2_SN[i])
probpost2_SN[i]<-num2_SN[i]/(num1_SN[i]+num2_SN[i])
}
class_SN<-vector()
for(i in 1:nrow(y)){
if(num1_SN[i] > num2_SN[i]){
class_SN[i]<-1
} else {
class_SN[i]<-2
}}
mcomp_SN<-matrix(c(probpost1_SN,probpost2_SN,class_SN,c2),ntes,4)
table(class_SN,c2)
n1=table(c2)[1]       #número de observações da classe 1
n2=table(c2)[2]     #número de observações da classe 2
bic1_SN=w1_SN$best.model$bic
bic2_SN=w2_SN$best.model$bic
conf_SN      <- matrix(c(table(class_SN,c2)),2,2)    #tabela de confundimento
n12_g1_SN <- conf_SN[1,2]  #n°. de observações da classe 1 alocada na classe 2
n21_g2_SN <- conf_SN[2,1] #n°. de observações da classe 2 alocada na classe 1
erro_total1_SN <- n12_g1_SN/n1 #erro total de classificação da classe 1 em 2
erro_total2_SN <- n21_g2_SN/n2  #erro total de classificação da classe 2 em 1
SN_g1<-cbind(k=k ,n1comp=ncomp1_SN, bic=bic1_SN,erro=erro_total1_SN )
SN_g2<-cbind(k=k,n2comp=ncomp2_SN, bic=bic2_SN, erro=erro_total2_SN)
write.table(SN_g1,paste(mydir,"SN_g1.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
write.table(SN_g2,paste(mydir,"SN_g2.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
mydir <- " "
```

```
#AJUSTANDO PELA T
##CLASSE 1
nu<-c(2,3,4,5,6,8,10)
i<-1
w1_T <- smsn.search(g1[,1:3],nu=nu[i], g.min = 1, g.max = 5, family = "t",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
bic_pre1 <- (w1_T $best.model)$bic
w_pre1 <- w1_T
nu_pre1<-nu[i]
for(i in 2:length(nu)){
    w1pos <- smsn.search(g1[,1:3],nu=nu[i], g.min = 1, g.max = 5, family = "t",
    criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
  bic_pos1 <- (w1pos$best.model)$bic
  w_pos1   <- w1pos
 nu_pos1   <- nu[i]
    if (bic_pre1 < bic_pos1){
         bic_aux1 <- bic_pre1
         w_aux1   <- w_pre1
         nu_aux1  <- nu_pre1
    }
    else {
         bic_aux1 <- bic_pos1
         w_aux1<- w_pos1
         nu_aux1  <- nu_pos1
   }
   bic_pre1<-bic_aux1
   w_pre1<-w_aux1
   nu_pre1 <-  nu_aux1
}
#CLASSE 2
nu<-c(2,3,4,5,6,8,10)
i<-1
w2_T <- smsn.search(g2[,1:3],nu=nu[i], g.min = 1, g.max = 5, family = "t",
 criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
bic_pre2 <- (w2_T $best.model)$bic
w_pre2 <- w2_T
nu_pre2<-nu[i]
for(i in 2:length(nu)){
    w2pos <- smsn.search(g2[,1:3],nu=nu[i], g.min = 1, g.max = 5, family = "t",
```

```
    criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
  bic_pos2 <- (w2pos$best.model)$bic
  w_pos2   <- w2pos
 nu_pos2   <- nu[i]
    if (bic_pre2 < bic_pos2){
         bic_aux2 <- bic_pre2
         w_aux2   <- w_pre2
         nu_aux2  <- nu_pre2
    }
    else {
         bic_aux2 <- bic_pos2
         w_aux2<- w_pos2
         nu_aux2  <- nu_pos2
   }
   bic_pre2<-bic_aux2
   w_pre2<-w_aux2
   nu_pre2 <-  nu_aux2
}
ncomp_T<-matrix(c(0,0,0),4,3)
iter2_T=iter1_T=ncomp1_T = ncomp2_T = vector()
w1_T<-w_pre1
ncomp1_T<- which.min(w1_T$criteria)
iter1_T<-w1_T$best.model$iter
w2_T<-w_pre2
ncomp2_T <- which.min(w2_T$criteria)
iter2_T<-w2_T$best.model$iter
p1_T<-matrix(c( w1_T$best.model$pii),ncomp1_T,1)
p2_T<-matrix(c( w2_T$best.model$pii),ncomp2_T,1)
mu_est1_T<-w1_T$best.model$mu
sigma_est1_T<-w1_T$best.model$Sigma
nu1_T<-w1_T$best.model$nu
#alocando as estimativas dos parâmetros do grupo 2 nas matrizes
mu_est2_T<-w2_T$best.model$mu
sigma_est2_T<-w2_T$best.model$Sigma
nu2_T<-w2_T$best.model$nu
probpost1_T=probpost2_T=num1_T=num2_T=vector()
sf1_T = pf1_T = f1_T = matrix(rep(0,ncomp1_T),ntes,ncomp1_T)
sf2_T = pf2_T = f2_T = matrix(rep(0,ncomp2_T),ntes,ncomp2_T)
#CLASSE 1
```

```
for(j in 1:ncomp1_T){
for(i in 1:nrow(y)){
f1_T[i,j]<-dmvt(y[i,1:3], delta=mu_est1_T[[j]],
    sigma=sigma_est1_T[[j]]%*%sigma_est1_T[[j]], df = nu1_T, log = F)
pf1_T[i,j]<-p1_T[j]*f1_T[i,j]
}}
sf1_T<-apply(pf1_T, 1, sum)
num1_T<-sf1_T*(nrow(g1)/ntre)
#CLASSE 2
for(j in 1:ncomp2_T){
for(i in 1:nrow(y)){
f2_T[i,j]<-dmvt(y[i,1:3], delta=mu_est2_T[[j]],
        sigma=sigma_est2_T[[j]]%*%sigma_est2_T[[j]], df = nu2_T, log = F)
pf2_T[i,j]<-p2_T[j]*f2_T[i,j]
}}
sf2_T<-apply(pf2_T, 1, sum)
num2_T<-sf2_T*(nrow(g2)/ntre)
for(i in 1:nrow(y)){
probpost1_T[i]<-num1_T[i]/(num1_T[i]+num2_T[i])
probpost2_T[i]<-num2_T[i]/(num1_T[i]+num2_T[i])
}
class_T<-vector()
for(i in 1:nrow(y)){
if(num1_T[i] > num2_T[i]){

class_T[i]<-1
} else {
class_T[i]<-2
}}
mcomp_T<-matrix(c(probpost1_T,probpost2_T,class_T,c2),ntes,4)
table(class_T,c2)
n1=table(c2)[1]
n2=table(c2)[2]
bic1_T=w1_T$best.model$bic
bic2_T=w2_T$best.model$bic
conf_T      <- matrix(c(table(class_T,c2)),2,2)    #tabela de confundimento
n12_g1_T <- conf_T[1,2]   #n°. de observações da classe 1 alocada na classe 2
n21_g2_T <- conf_T[2,1] #n°. de observações da classe 2 alocada na classe 1
erro_total1_T <- n12_g1_T/n1 #erro total de classificação da classe 1 em 2
```

```
erro_total2_T <- n21_g2_T/n2  #erro total de classificação da classe 2 em 1
T_g1<-cbind(k=k,nu=nu_pre1, n1comp=ncomp1_T, bic=bic1_T,erro=erro_total1_T )
T_g2<-cbind(k=k,nu=nu_pre2,n2comp=ncomp2_T, bic=bic2_T, erro=erro_total2_T)
write.table(T_g1,paste(mydir,"T_g1.txt",sep=""),append=TRUE,row.names=F,col.names=F
write.table(T_g2,paste(mydir,"T_g2.txt",sep=""),append=TRUE,row.names=F,col.names=F
}
mydir <- " "
#AJUSTANDO PELA SKEW-T
##CLASSE 1
nu<-c(2,3,4,5,6,8,10)
i<-1
w1_ST <- smsn.search(g1[,1:2],nu=nu[i], g.min = 1, g.max = 5, family = "Skew.t",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
bic_pre1 <- (w1_ST $best.model)$bic
w_pre1 <- w1_ST
nu_pre1<-nu[i]
for(i in 2:length(nu)){
    w1pos <- smsn.search(g1[,1:2],nu=nu[i], g.min = 1, g.max = 5,
    family = "Skew.t", criteria = "bic", error = 0.0001,
    iter.max = 100, calc.im = FALSE)
  bic_pos1 <- (w1pos$best.model)$bic
  w_pos1   <- w1pos
 nu_pos1   <- nu[i]
    if (bic_pre1 < bic_pos1){
         bic_aux1 <- bic_pre1
         w_aux1   <- w_pre1
         nu_aux1  <- nu_pre1
    }
    else {
         bic_aux1 <- bic_pos1
         w_aux1<- w_pos1
         nu_aux1  <- nu_pos1
   }
   bic_pre1<-bic_aux1
   w_pre1<-w_aux1
   nu_pre1 <-  nu_aux1
   }
#CLASSE 2
nu<-c(2,3,4,5,6,8,10)
```

```
i<-1
w2_ST <- smsn.search(g2[,1:2],nu=nu[i], g.min = 1, g.max = 5, family = "Skew.t",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
bic_pre2 <- (w2_ST $best.model)$bic
w_pre2 <- w2_ST
nu_pre2<-nu[i]
for(i in 2:length(nu)){
    w2pos <- smsn.search(g2[,1:2],nu=nu[i], g.min = 1, g.max = 5,
    family = "Skew.t", criteria = "bic", error = 0.0001,
    iter.max = 100, calc.im = FALSE)
  bic_pos2 <- (w2pos$best.model)$bic
  w_pos2   <- w2pos
 nu_pos2   <- nu[i]
    if (bic_pre2 < bic_pos2){
         bic_aux2 <- bic_pre2
         w_aux2   <- w_pre2
         nu_aux2  <- nu_pre2
    }
    else {
         bic_aux2 <- bic_pos2
         w_aux2<- w_pos2
         nu_aux2  <- nu_pos2
   }
   bic_pre2<-bic_aux2
   w_pre2<-w_aux2
   nu_pre2 <-  nu_aux2
   }
ncomp_ST<-matrix(c(0,0,0),4,3)
iter2_ST=iter1_ST=ncomp1_ST = ncomp2_ST = vector()
w1_ST<-w_pre1
ncomp1_ST<- which.min(w1_ST$criteria)
iter1_ST<-w1_ST$best.model$iter
w2_ST<-w_pre2
ncomp2_ST <- which.min(w2_ST$criteria)
iter2_ST<-w2_ST$best.model$iter
p1_ST<-matrix(c( w1_ST$best.model$pii),ncomp1_ST,1)
p2_ST<-matrix(c( w2_ST$best.model$pii),ncomp2_ST,1)
#alocando as estimativas dos parâmetros da classe 1 nas matrizes
mu_est1_ST<-w1_ST$best.model$mu
```

```
sigma_est1_ST<-w1_ST$best.model$Sigma
nu1_ST<-w1_ST$best.model$nu
shape_est1_ST<-w1_ST$best.model$shape
#alocando as estimativas dos parâmetros da classe 2 nas matrizes
mu_est2_ST<-w2_ST$best.model$mu
sigma_est2_ST<-w2_ST$best.model$Sigma
nu2_ST<-w2_ST$best.model$nu
shape_est2_ST<-w2_ST$best.model$shape
probpost1_ST=probpost2_ST=num1_ST=num2_ST=vector()
sf1_ST = pf1_ST = f1_ST = matrix(rep(0,ncomp1_ST),ntes,ncomp1_ST)
sf2_ST = pf2_ST = f2_ST = matrix(rep(0,ncomp2_ST),ntes,ncomp2_ST)
#CLASSE 1
for(j in 1:ncomp1_ST){
for(i in 1:nrow(y)){
f1_ST[i,j]<-dmst(y[i,1:2], xi=mu_est1_ST[[j]],
    Omega=sigma_est1_ST[[j]]%*%sigma_est1_ST[[j]],
    alpha=shape_est1_ST[[j]], df=nu1_ST)
pf1_ST[i,j]<-p1_ST[j]*f1_ST[i,j]
}}
sf1_ST<-apply(pf1_ST, 1, sum)
num1_ST<-sf1_ST*(nrow(g1)/ntre)
#CLASSE 2
for(j in 1:ncomp2_ST){
for(i in 1:nrow(y)){
f2_ST[i,j]<-dmst(y[i,1:2], xi=mu_est2_ST[[j]],
       Omega=sigma_est2_ST[[j]]%*%sigma_est2_ST[[j]],
       alpha=shape_est2_ST[[j]], df=nu2_ST)
pf2_ST[i,j]<-p2_ST[j]*f2_ST[i,j]
}}
sf2_ST<-apply(pf2_ST, 1, sum)
num2_ST<-sf2_ST*(nrow(g2)/ntre)
for(i in 1:nrow(y)){
probpost1_ST[i]<-num1_ST[i]/(num1_ST[i]+num2_ST[i])
probpost2_ST[i]<-num2_ST[i]/(num1_ST[i]+num2_ST[i])
}
class_ST<-vector()
for(i in 1:nrow(y)){
if(num1_ST[i] > num2_ST[i]){
class_ST[i]<-1
```

```
} else {
class_ST[i]<-2
}}
mcomp_ST<-matrix(c(probpost1_ST,probpost2_ST,class_ST,c2),ntes,4)
table(class_ST,c2)
n1=table(c2)[1]       #número de observações da classe 1
n2=table(c2)[2]     #número de observações da classe 2
bic1_ST=w1_ST$best.model$bic
bic2_ST=w2_ST$best.model$bic
conf_ST      <- matrix(c(table(class_ST,c2)),2,2)    #tabela de confundimento
n12_g1_ST <- conf_ST[1,2] #n°. de observações da classe 1 alocada na classe 2
n21_g2_ST <- conf_ST[2,1]  #n°. de observações da classe 2 alocada na classe 1
erro_total1_ST <- n12_g1_ST/n1 #erro total de classificação da classe 1 em 2
erro_total2_ST <- n21_g2_ST/n2  #erro total de classificação da classe 2 em 1
ST_g1<-cbind(k=k,nu=nu_pre1, n1comp=ncomp1_ST, bic=bic1_ST,erro=erro_total1_ST)
ST_g2<-cbind(k=k,nu=nu_pre2, n2comp=ncomp2_ST, bic=bic2_ST, erro=erro_total2_ST)
write.table(ST_g1,paste(mydir,"ST_g1.txt",sep=""),append=TRUE,row.names=F,
col.names=F) #guardando o objeto da classe 1
write.table(ST_g2,paste(mydir,"ST_g2.txt",sep=""),append=TRUE,row.names=F,
col.names=F)  #guardando o objeto da classe 2
}
```

A.2 Dados Reais

```
library(mvtnorm)
library(mixsmsn)
library(MASS)
library(sn)
#----------- Leitura dos dados - Caranguejos Leptograpsus ---------------------
dados=read.table("Caranguejos1.txt",h=T)
attach(dados)
n=nrow(dados)
g1=dados[dados$sp==1,]
g2=dados[dados$sp==2,]
indices = matrix(1:200,40,5,byrow = T)
for (l in 2:40){
     print(as.vector(indices[l,]))
```

```
     tre=dados[-as.vector(indices[l,]),]
     teste=dados[as.vector(indices[l,]),]
g1=tre[tre$sp==1,]  #treino
g2=tre[tre$sp==2,]  #treino
g1obs<-teste[teste$sp==1,]    #teste
g2obs<-teste[teste$sp==2,]     #teste
vari=4:8 #colunas das variáveis, na matriz de dados
#------- Leitura dos dados - Wisconsin Breast Cancer Diagnóstico (WDBC)---------
x=read.table("wdbc_dados.txt",h=F)
attach(x)
n=nrow(x)
dadosc=x[,3:32] #matriz somente das variáveis
cp=princomp(dadosc) #componentes principais
scp=summary(cp)  #sumário das componentes
scores=scp$scores #escores
dados=matrix(c(x$V2,scores[,1],scores[,2],scores[,3],scores[,4]),569,5)
g1=dados[dados[,1]==1,]
g2=dados[dados[,1]==2,]
vari=2:5  #colunas das variáveis, na matriz de dados
#------------------------------- k-fold, para k=6 -----------------------------
indices = matrix(1:570,6,95,byrow = T)
for (l in 1:5){
    print(as.vector(indices[l,]))
    #l=c(476:569)
    tre=dados[-as.vector(indices[l,]),]
    #tre=dados[-l,]
    teste=dados[as.vector(indices[l,]),]    #
    #teste=dados[l,]
    ntes=nrow(teste)
#------------------- separando as classes no conjunto de treinamento -----------
g1=tre[tre[,1]==1,]     #treino
g2=tre[tre[,1]==2,]     #treino
#--------------- separando as classes no conjunto de teste ---------------------
g1obs<-teste[teste[,1]==1,]    #teste
g2obs<-teste[teste[,1]==2,]     #teste
#----------- Leitura dos dados - Baciloscopia ----------------------
x=read.table('tb.txt',header=T)
head(x, n=10L)
attach(x)
```

```
n=nrow(x) #número total de observações
ca=5 #número de parte do k-fold (5, 8 e 10)
linhas=4800/ca #número de linhas das matrizes de treino e teste (bacilo+fundo)
nv=4 #número de variáveis 4,5,6,7,8
##SELECIONANDO AS VARIÁVEIS
colunas1=c(20,23,25,29,31,33)       # 4 variáveis
#colunas2=c(7,20,23,25,29,31,33)     # 5 variáveis
#colunas3=c(7,14,20,23,25,29,31,33)  # 6 variáveis
#colunas4=c(7,14,20,23,25,27,29,31,33)    # 7 variáveis
#colunas5=c(7,10,14,20,23,25,27,29,31,33) # 8 variáveis
##SEPARANDO BACILO E FUNDO
dados=x[,colunas1]#conjunto de dados com as variáveis selecionadas
fundo=dados[dados[,6]=="F",] #matriz com os dados de fundo  2400 obs
bacilo=dados[dados[,6]=="B",]#matriz com os dados de bacilo 2400 obs
head(fundo)
head(bacilo)
#SEPARANDO AS PARTES PARA A VALIDAÇÃO (INDICES)
f_K_fold <- function(Nobs,K){
    rs <- runif(Nobs)
    id <- seq(Nobs)[order(rs)]
    k <- as.integer(Nobs*seq(1,K-1)/K)
    k <- matrix(c(0,rep(k,each=2),Nobs),ncol=2,byrow=TRUE)
    k[,1] <- k[,1]+1
    l <- lapply(seq.int(K),function(x,k,d)
                list(train=d[!(seq(d) %in% seq(k[x,1],k[x,2]))],
                     test=d[seq(k[x,1],k[x,2])]),k=k,d=id)
   return(l)
}
partes=f_K_fold(2400,ca) # separando em ca= 5, 8 e 10 partes
#Separando as partes para fundo e bacilo
testef=treinof=testeb=treinob=listaf=listab=list()
 for(k in 1:ca){
 testef[[k]]  <- fundo[-partes[[k]][[1]],]
 treinof[[k]] <- fundo[-partes[[k]][[2]],]
 testeb[[k]]  <- bacilo[-partes[[k]][[1]],]
 treinob[[k]] <- bacilo[-partes[[k]][[2]],]
listaf[[k]]=list(treinof[[k]],testef[[k]])
listab[[k]]=list(treinob[[k]],testeb[[k]])
}
```

```
listabf=list(listab,listaf)
#Definir conjunto de treino (tre) que são as observações menos 5;
for(l in 1:ca){
#selecionando fundo e bacilo[[1/2]],da i-ésima parte[[i]],somente o treino [[1]]
    tre=rbind(listabf[[1]][[l]][[1]],listabf[[2]][[l]][[1]])
#selecionando fundo e bacilo[[1/2]],da i-ésima parte[[i]], somente o teste[[2]]
    teste=rbind(listabf[[1]][[l]][[2]],listabf[[2]][[l]][[2]])
    ntes=nrow(teste)
    ntre=nrow(tre)
#SEPARANDO CONJ. DE TREINAMENTO E TESTE - ÍMPARES E PARES
indices=1:2400
pares=indices[indices %% 2 == 0 ]
impares=indices[indices %% 2 != 0 ]
testef=fundo[pares,]
treinof=fundo[impares,]
testeb=bacilo[pares,]
treinob=bacilo[impares,]
tre=rbind(treinof,treinob) #selecionando fundo e bacilo [[1/2]], da i-ésima parte [
teste=rbind(testef,testeb) #selecionando fundo e bacilo [[1/2]], da i-ésima parte [
ntes=nrow(teste)
ntre=nrow(tre)
#----------------------------- separando grupos no conjunto de treino ----------
g1=tre[tre[,6]=="B",]     #treino
g2=tre[tre[,6]=="F",]     #treino
#----------------------------- separando grupos das observações de teste -------
g1obs<-teste[teste[,6]=="B",]    #teste
g2obs<-teste[teste[,6]=="F",]     #teste
vari=1:nv #colunas das variáveis, na matriz de dados
#---------------------- ajustando pelas distribuições -------------------------
mydir <- " "
#AJUSTANDO PELA NORMAL
iter2_N=iter1_N=ncomp1_N = ncomp2_N = vector()
#usando treino
w1_N<- smsn.search(g1[,vari],nu=2, g.min = 1, g.max = 5,family="Normal",
criteria<- "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
ncomp1_N <- which.min(w1_N$criteria)
iter1_N<- w1_N$best.model$iter
#ajustando as famílias e o número de componentes indicado, para a classe 2
w2_N <-smsn.search(g2[,vari],nu=2,g.min = 1,g.max = 5,family = "Normal",
```

```
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
ncomp2_N   <- which.min(w2_N$criteria)
iter2_N<-w2_N$best.model$iter
p1_N<-matrix(c( w1_N$best.model$pii),ncomp1_N,1)
p2_N<-matrix(c( w2_N$best.model$pii),ncomp2_N,1)
#alocando as estimativas dos parâmetros da classe 1 nas matrizes
mu_est1_N     <-w1_N$best.model$mu
sigma_est1_N <-w1_N$best.model$Sigma
#alocando as estimativas dos parâmetros da classe 2 nas matrizes
mu_est2_N     <-w2_N$best.model$mu
sigma_est2_N <-w2_N$best.model$Sigma
#------------------- DENSIDADES PARA CLASSIFICAÇÃO ----------------------------
probpost1_N = probpost2_N=num1_N=num2_N=vector()
sf1_N = pf1_N = f1_N = matrix(rep(0,ncomp1_N),ntes,ncomp1_N)
sf2_N = pf2_N = f2_N = matrix(rep(0,ncomp2_N),ntes,ncomp2_N)
#Utilizando o conjunto teste, calcular as densidades para
#o classificador de Bayes;
#CLASSE 1
for(j in 1:ncomp1_N){
for(i in 1:nrow(teste)){
f1_N[i,j]<-dmvnorm(teste[i,vari],mean=mu_est1_N[[j]],
    sigma=sigma_est1_N[[j]]%*%sigma_est1_N[[j]])
pf1_N[i,j]<-p1_N[j]*f1_N[i,j]
}}
sf1_N<-apply(pf1_N, 1, sum)
num1_N<-sf1_N*(nrow(g1)/nrow(tre))
##CLASSE 2
for(j in 1:ncomp2_N){
for(i in 1:nrow(teste)){
f2_N[i,j]<-dmvnorm(teste[i,vari], mean=mu_est2_N[[j]],
        sigma=sigma_est2_N[[j]]%*%sigma_est2_N[[j]])
pf2_N[i,j]<-p2_N[j]*f2_N[i,j]
}}
sf2_N<-apply(pf2_N, 1, sum)
num2_N<-sf2_N*(nrow(g2)/nrow(tre))
#---------------- PROBABILIDADES POSTERIORIS PARA CADA CLASSE
for(i in 1:nrow(teste)){
probpost1_N[i]<-num1_N[i]/(num1_N[i]+num2_N[i])
probpost2_N[i]<-num2_N[i]/(num1_N[i]+num2_N[i])
```

```
}
#Comparar as densidades (classificador) e classificar
class_N<-vector()
for(i in 1:nrow(teste)){
if(num1_N[i] > num2_N[i]){
class_N[i]<-1
} else {
class_N[i]<-2
}}
matriz[,l]=class_N
comp[l,]=cbind(ncomp1_N,ncomp2_N)
}
n1=nrow(g1obs)       #número de observações da classe 1
n2=nrow(g2obs)     #número de observações da classe 2
bic1_N=w1_N$best.model$bic
bic2_N=w2_N$best.model$bic
N_g1<-cbind(l=l ,n1comp=ncomp1_N, bic=bic1_N)
N_g2<-cbind(l=l,n2comp=ncomp2_N, bic=bic2_N)
write.table(N_g1,paste(mydir,"N_g1.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
write.table(N_g2,paste(mydir,"N_g2.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
matriz=matrix(c(class_N),ntes,1)
print(matriz)
}
mydir <- " "
#AJUSTANDO PELA SKEW-NORMAL
iter2_SN=iter1_SN=ncomp1_SN = ncomp2_SN = vector()
w1_SN<-smsn.search(g1[,vari],nu=2, g.min = 1, g.max = 5, family = "Skew.normal",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
ncomp1_SN<- which.min(w1_SN$criteria)
iter1_SN<-w1_SN$best.model$iter
w2_SN<-smsn.search(g2[,vari],nu=2, g.min = 1, g.max = 5, family = "Skew.normal",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
ncomp2_SN <- which.min(w2_SN$criteria)
iter2_SN<-w2_SN$best.model$iter
p1_SN<-matrix(c(w1_SN$best.model$pii),ncomp1_SN,1)
p2_SN<-matrix(c(w2_SN$best.model$pii),ncomp2_SN,1)
mu_est1_SN<-w1_SN$best.model$mu
```

```
sigma_est1_SN<-w1_SN$best.model$Sigma
shape_est1_SN<-w1_SN$best.model$shape
mu_est2_SN<-w2_SN$best.model$mu
sigma_est2_SN<-w2_SN$best.model$Sigma
shape_est2_SN<-w2_SN$best.model$shape
probpost1_SN=probpost2_SN=num1_SN=num2_SN=vector()
sf1_SN = pf1_SN = f1_SN = matrix(rep(0,ncomp1_SN),ntes,ncomp1_SN)
sf2_SN = pf2_SN = f2_SN = matrix(rep(0,ncomp2_SN),ntes,ncomp2_SN)
##GRUPO 1
for(j in 1:ncomp1_SN){
for(i in 1:nrow(teste)){
f1_SN[i,j]<-dmsn(teste[i,vari], xi = mu_est1_SN[[j]],
    Omega=sigma_est1_SN[[j]]%*%sigma_est1_SN[[j]], alpha=shape_est1_SN[[j]])
pf1_SN[i,j]<-p1_SN[j]*f1_SN[i,j]
  }}
sf1_SN<-apply(pf1_SN, 1, sum)
num1_SN<-sf1_SN*(nrow(g1)/nrow(tre))
##GRUPO 2
for(j in 1:ncomp2_SN){
for(i in 1:nrow(teste)){
f2_SN[i,j]<-dmsn(teste[i,vari], xi = mu_est2_SN[[j]],
        Omega=sigma_est2_SN[[j]]%*%sigma_est2_SN[[j]], alpha=shape_est2_SN[[j]])
pf2_SN[i,j]<-p2_SN[j]*f2_SN[i,j]
    }}
sf2_SN<-apply(pf2_SN, 1, sum)
num2_SN<-sf2_SN*(nrow(g2)/nrow(tre))
#-------------------------- PROBABILIDADES POSTERIORIS PARA CADA GRUPO ---------
for(i in 1:nrow(teste)){
probpost1_SN[i]<-num1_SN[i]/(num1_SN[i]+num2_SN[i])
probpost2_SN[i]<-num2_SN[i]/(num1_SN[i]+num2_SN[i])
}
class_SN<-vector()
for(i in 1:nrow(teste)){
if(num1_SN[i] > num2_SN[i]){
class_SN[i]<-1
} else {
class_SN[i]<-2
}}
matriz[,l]=class_SN
```

```
comp[l,]=cbind(ncomp1_SN,ncomp2_SN)
}
n1=nrow(g1obs)       #número de observações do grupo 1
n2=nrow(g2obs)     #número de observações do grupo 2
bic1_SN=w1_SN$best.model$bic
bic2_SN=w2_SN$best.model$bic
SN_g1<-cbind(l=l,n1comp=ncomp1_SN, bic=bic1_SN)
SN_g2<-cbind(l=l,n2comp=ncomp2_SN, bic=bic2_SN)
write.table(SN_g1,paste(mydir,"SN_g1.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
write.table(SN_g2,paste(mydir,"SN_g2.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
matriz=matrix(c(class_SN),ntes,1)
print(matriz)
}
mydir <- " "
#AJUSTANDO PELA T
#CLASSE 1
nu<-c(2,3,4,5,6,8,10)
i<-1
w1_T <- smsn.search(g1[,vari],nu=nu[i], g.min = 1, g.max = 5, family = "t",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
bic_pre1 <- (w1_T $best.model)$bic
w_pre1 <- w1_T
nu_pre1<-nu[i]
for(i in 2:length(nu)){
    w1pos <- smsn.search(g1[,vari],nu=nu[i], g.min = 1, g.max = 4, family = "t",
    criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
  bic_pos1 <- (w1pos$best.model)$bic
  w_pos1   <- w1pos
 nu_pos1   <- nu[i]
    if (bic_pre1 < bic_pos1){
         bic_aux1 <- bic_pre1
         w_aux1   <- w_pre1
         nu_aux1  <- nu_pre1
    }
    else {
         bic_aux1 <- bic_pos1
         w_aux1<- w_pos1
```

```
          nu_aux1  <- nu_pos1
   }
   bic_pre1<-bic_aux1
   w_pre1<-w_aux1
   nu_pre1 <-  nu_aux1
}
#CLASSE 2
nu<-c(2,3,4,5,6,8,10)
i<-1
w2_T <- smsn.search(g2[,vari],nu=nu[i], g.min = 1, g.max = 5, family = "t",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
bic_pre2 <- (w2_T $best.model)$bic
w_pre2 <- w2_T
nu_pre2<-nu[i]
for(i in 2:length(nu)){
    w2pos <- smsn.search(g2[,vari],nu=nu[i], g.min = 1, g.max = 5, family = "t",
    criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
  bic_pos2 <- (w2pos$best.model)$bic
  w_pos2   <- w2pos
 nu_pos2   <- nu[i]
    if (bic_pre2 < bic_pos2){
         bic_aux2 <- bic_pre2
         w_aux2   <- w_pre2
         nu_aux2  <- nu_pre2
    }
    else {
         bic_aux2 <- bic_pos2
         w_aux2<- w_pos2
         nu_aux2  <- nu_pos2
   }
   bic_pre2<-bic_aux2
   w_pre2<-w_aux2
   nu_pre2 <-  nu_aux2
}
ncomp_T<-matrix(c(0,0,0),4,3)
iter2_T=iter1_T=ncomp1_T = ncomp2_T = vector()
w1_T<-w_pre1
ncomp1_T<- which.min(w1_T$criteria)
iter1_T<-w1_T$best.model$iter
```

```
w2_T<-w_pre2
ncomp2_T <- which.min(w2_T$criteria)
iter2_T<-w2_T$best.model$iter
p1_T<-matrix(c( w1_T$best.model$pii),ncomp1_T,1)
p2_T<-matrix(c( w2_T$best.model$pii),ncomp2_T,1)
mu_est1_T<-w1_T$best.model$mu
sigma_est1_T<-w1_T$best.model$Sigma
nu1_T<-w1_T$best.model$nu
mu_est2_T<-w2_T$best.model$mu
sigma_est2_T<-w2_T$best.model$Sigma
nu2_T<-w2_T$best.model$nu
probpost1_T=probpost2_T=num1_T=num2_T=vector()
sf1_T = pf1_T = f1_T = matrix(rep(0,ncomp1_T),ntes,ncomp1_T)
sf2_T = pf2_T = f2_T = matrix(rep(0,ncomp2_T),ntes,ncomp2_T)
#CLASSE 1
for(j in 1:ncomp1_T){
   for(i in 1:nrow(teste)){
f1_T[i,j]<-dmvt(teste[i,vari], delta=mu_est1_T[[j]],
    sigma=sigma_est1_T[[j]]%*%sigma_est1_T[[j]], df = nu1_T, log = F)
    pf1_T[i,j]<-p1_T[j]*f1_T[i,j]
}}
sf1_T<-apply(pf1_T, 1, sum)
num1_T<-sf1_T*(nrow(g1)/nrow(tre))
#CLASSE 2
for(j in 1:ncomp2_T){
for(i in 1:nrow(teste)){
  f2_T[i,j]<-dmvt(teste[i,vari], delta=mu_est2_T[[j]],
      sigma=sigma_est2_T[[j]]%*%sigma_est2_T[[j]], df = nu2_T, log = F)
      pf2_T[i,j]<-p2_T[j]*f2_T[i,j]
}}
sf2_T<-apply(pf2_T, 1, sum)
num2_T<-sf2_T*(nrow(g2)/nrow(tre))
for(i in 1:nrow(teste)){
probpost1_T[i]<-num1_T[i]/(num1_T[i]+num2_T[i])
probpost2_T[i]<-num2_T[i]/(num1_T[i]+num2_T[i])
}
class_T<-vector()
for(i in 1:nrow(teste)){
if(num1_T[i] > num2_T[i]){
```

```
class_T[i]<-1
} else {
class_T[i]<-2
}}
matriz[,l]=class_T
comp[l,]=cbind(ncomp1_ST,ncomp2_T)
}
n1=nrow(g1obs)       #número de observações do grupo 1
n2=nrow(g2obs)     #número de observações do grupo 2
bic1_T=w1_T$best.model$bic
bic2_T=w2_T$best.model$bic
T_g1<-cbind(l=l,nu=nu_pre1, n1comp=ncomp1_T, bic=bic1_T)
T_g2<-cbind(l=l,nu=nu_pre2,n2comp=ncomp2_T, bic=bic2_T)
write.table(T_g1,paste(mydir,"T_g1.txt",sep=""),append=TRUE,row.names=F,
col.names=F) #guardando o objeto do grupo 1
write.table(T_g2,paste(mydir,"T_g2.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
print(class_T) #preciso desse vetor também (vetor de clasificação)
matriz=matrix(c(class_T),ntes,1)
matriz
}
mydir <- " "
#AJUSTANDO PELA SKEW-T
#CLASSE 1
 nu<-c(2,3,4,5,6,8,10)
i<-1
w1_ST <- smsn.search(g1[,vari],nu=nu[i], g.min = 1, g.max = 5,
family = "Skew.t", criteria = "bic", error = 0.0001, iter.max = 100,
calc.im = FALSE)
bic_pre1 <- (w1_ST $best.model)$bic
w_pre1 <- w1_ST
nu_pre1<-nu[i]
for(i in 2:length(nu)){
    w1pos <- smsn.search(g1[,vari],nu=nu[i], g.min = 1, g.max = 4,
    family = "Skew.t", criteria = "bic", error = 0.0001, iter.max = 100,
    calc.im = FALSE)
  bic_pos1 <- (w1pos$best.model)$bic
  w_pos1   <- w1pos
 nu_pos1   <- nu[i]
```

```
    if (bic_pre1 < bic_pos1){
         bic_aux1 <- bic_pre1
         w_aux1   <- w_pre1
         nu_aux1  <- nu_pre1
    }
    else {
         bic_aux1 <- bic_pos1
         w_aux1<- w_pos1
         nu_aux1  <- nu_pos1
   }
   bic_pre1<-bic_aux1
   w_pre1<-w_aux1
   nu_pre1 <-  nu_aux1
}
#CLASSE 2
nu<-c(2,3,4,5,6,8,10)
i<-1
w2_ST <- smsn.search(g2[,vari],nu=nu[i], g.min = 1, g.max = 5, family = "Skew.t",
criteria = "bic", error = 0.0001, iter.max = 100, calc.im = FALSE)
bic_pre2 <- (w2_ST $best.model)$bic
w_pre2 <- w2_ST
nu_pre2<-nu[i]
for(i in 2:length(nu)){
    w2pos<-smsn.search(g2[,vari],nu=nu[i], g.min = 1, g.max = 5,
    family = "Skew.t", criteria = "bic", error = 0.0001, iter.max = 100,
    calc.im = FALSE)
  bic_pos2 <- (w2pos$best.model)$bic
  w_pos2   <- w2pos
 nu_pos2   <- nu[i]
    if (bic_pre2 < bic_pos2){
         bic_aux2 <- bic_pre2
         w_aux2   <- w_pre2
         nu_aux2  <- nu_pre2
    }
    else {
         bic_aux2 <- bic_pos2
         w_aux2<- w_pos2
         nu_aux2  <- nu_pos2
   }
```

```
   bic_pre2<-bic_aux2
   w_pre2<-w_aux2
   nu_pre2 <-  nu_aux2
}
ncomp_ST<-matrix(c(0,0,0),4,3)
iter2_ST=iter1_ST=ncomp1_ST = ncomp2_ST = vector()
w1_ST<-w_pre1
ncomp1_ST<- which.min(w1_ST$criteria)
iter1_ST<-w1_ST$best.model$iter
w2_ST<-w_pre2
ncomp2_ST <- which.min(w2_ST$criteria)
iter2_ST<-w2_ST$best.model$iter
p1_ST<-matrix(c( w1_ST$best.model$pii),ncomp1_ST,1)
p2_ST<-matrix(c( w2_ST$best.model$pii),ncomp2_ST,1)
mu_est1_ST<-w1_ST$best.model$mu
sigma_est1_ST<-w1_ST$best.model$Sigma
nu1_ST<-w1_ST$best.model$nu
shape_est1_ST<-w1_ST$best.model$shape
mu_est2_ST<-w2_ST$best.model$mu
sigma_est2_ST<-w2_ST$best.model$Sigma
nu2_ST<-w2_ST$best.model$nu
shape_est2_ST<-w2_ST$best.model$shape
probpost1_ST=probpost2_ST=num1_ST=num2_ST=vector()
sf1_ST = pf1_ST = f1_ST = matrix(rep(0,ncomp1_ST),ntes,ncomp1_ST)
sf2_ST = pf2_ST = f2_ST = matrix(rep(0,ncomp2_ST),ntes,ncomp2_ST)
##GRUPO 1
for(j in 1:ncomp1_ST){
for(i in 1:nrow(teste)){
f1_ST[i,j]<-dmst(teste[i,vari], xi=mu_est1_ST[[j]],
    Omega=sigma_est1_ST[[j]]%*%sigma_est1_ST[[j]],
    alpha=shape_est1_ST[[j]], df=nu1_ST)
pf1_ST[i,j]<-p1_ST[j]*f1_ST[i,j]
}}
sf1_ST<-apply(pf1_ST, 1, sum)
num1_ST<-sf1_ST*(nrow(g1)/nrow(tre))
#CLASSE 2
for(j in 1:ncomp2_ST){
for(i in 1:nrow(teste)){
f2_ST[i,j]<-dmst(teste[i,vari], xi=mu_est2_ST[[j]],
```

```
      Omega=sigma_est2_ST[[j]]%*%sigma_est2_ST[[j]],
      alpha=shape_est2_ST[[j]], df=nu2_ST)
pf2_ST[i,j]<-p2_ST[j]*f2_ST[i,j]
    }}
sf2_ST<-apply(pf2_ST, 1, sum)
num2_ST<-sf2_ST*(nrow(g2)/nrow(tre))
for(i in 1:nrow(teste)){
probpost1_ST[i]<-num1_ST[i]/(num1_ST[i]+num2_ST[i])
probpost2_ST[i]<-num2_ST[i]/(num1_ST[i]+num2_ST[i])
}
class_ST<-vector()
for(i in 1:nrow(teste)){
if(num1_ST[i] > num2_ST[i]){
class_ST[i]<-1
} else {

class_ST[i]<-2
}}
matriz[,l]=class_ST
comp[l,]=cbind(ncomp1_ST,ncomp2_ST)
}
n1=nrow(g1obs)       #número de observações do grupo 1
n2=nrow(g2obs)     #número de observações do grupo 2
bic1_ST=w1_ST$best.model$bic
bic2_ST=w2_ST$best.model$bic
ST_g1<-cbind(l=l,nu=nu_pre1, n1comp=ncomp1_ST, bic=bic1_ST)
ST_g2<-cbind(l=l,nu=nu_pre2, n2comp=ncomp2_ST, bic=bic2_ST)
write.table(ST_g1,paste(mydir,"ST_g1.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
write.table(ST_g2,paste(mydir,"ST_g2.txt",sep=""),append=TRUE,row.names=F,
col.names=F)
print(class_ST)
matriz=matrix(c(class_ST),ntes,1)
print(matriz)
}
```

Referências Bibliográficas

[1] Andrews, J. L. & McNicholas, P. D. (2012). Model–Based Clustering, Classification, and Discriminant Analysis via Mixtures of Multivariate *t*–Distributions. *Statistics and Computing*, **22**, 1021–1029.

[2] Arellano-Vale, R. B. & Azzalini, A. (2006). On the unification of families of skew–normal distributions. *scandj*, **33**, 561–574.

[3] Azzalini, A. (2005). The skew–normal distributions which includes the normal ones. *scandj*, **32**, 159–188.

[4] Basso, R. M. (2010). *Misturas Finitas de Misturas de Escala Skew-Normal*. Dissertação de Mestrado - Universidade Estadual de Campinas, São Paulo.

[5] Cabral, C. R. B., Lachos, V. H. & Prates, M. O. (2012). Multivariate mixture modeling using skew-normal independent distributions. *Computational Statistics and Data Analysis*, **56**, 126–142.

[6] Dempster, A. P., Laird, N. M. & Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society Series B*, **39**(1), 1–38.

[7] Filho, C. F. F. C., Levy, P. C., Xavier, C. M., Costa, M. G. F., Fujimoto, L. B. M. & Jose, J. I. N. S. (2012). Mycobacterium tuberculosis recognition with conventional microscopy. *34th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*.

[8] Fraley, C. & Raftery, A. E. (2002). Model-Based Clustering, Discriminant Analysis, and Density Estimation. *Journal of the American Statistical Association*, **97**(458), 611–631.

[9] Hartigan, J. A. & Wong, M. A. (1979). A K–means clustering algorithm. *Applied Statistics*, **28**, 100–108.

[10] Hastie, T., Tibshirani, R. & Friedman, J. (2009). *The Elements of Statistical Learning. Data Mining, Inference, and Prediction. Second Edition*. Springer, USA.

[11] Jain, A. K., Duin, R. P. W. & Mao, J. (2000). Statistical Pattern Recognition: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(1), 4–37.

[12] Johnson, R. A. & Wichern, D. W. (2007). *Applied Multivariate Statistical Analysis. Sixth Edition*. Prentice Hall, USA.

[13] Lee, S. & McLachlan, G. J. (2011). On the fitting of mixtures of multivariate skew t-distributions via the EM algorithm. *arXiv: 1109.4706v[Stat.ME]*.

[14] Lee, S. & McLachlan, G. J. (2012). Finite mixtures of multivariate skew t-distributions: some recent and new results. *Statistics and Computing*, (DOI 10.1007/s11222-012-9362-4).

[15] Levy, P. C. (2012). *Reconhecimento e segmentação do Microbacterium Tuberculosis em Imagens de Microscopia de Campo Claro Utilizando as características de cor e o Algoritmo Backpropagation*. Dissertação de Mestrado – Universidade Federal do Amazonas, Amazonas.

[16] Levy, P. C., Xavier, C. M., Filho, C. F. F. C., Costa, M. G. F., Fujimoto, L. B. M. & Jose, J. I. N. S. (2012). Segmentação do mycobacterium tuberculosis através de novas técnicas de classificação de pixel. *Anais do Workshop de Informática Médica - WIM*.

[17] Liu, C. & Rubin, D. B. (1994). The ECM algorithm: A simple extension of EM and ECM with faster monotone convergence. *biotka*, **81**, 633–648.

[18] Maugis, C., Celeux, G. & Martin-Magniette, M.-L. (2010). Variable selection in model-based discriminant analysis. *INRIA Rapport de Recherche* N^{o}792.

[19] McLachlan, G. & Peel, D. (2000). *Finite Mixture Models*. John Wiley & Sons, New York.

[20] McLachlan, G. J. (2004). *Discriminant Analysis and Statistical Pattern Recognition*. Wiley, New York.

[21] McLachlan, G. J. & Krishnan, T. (2008). *The EM Algorithm and Extensions. Second Edition*. Wiley, USA.

[22] Pereira, J. R. G. (2001). *Misturas Finitas de Densidades com aplicações em Reconhecimento Estatístico de Padrões*. Tese de Doutorado - Universidade Estadual de Campinas, São Paulo.

[23] Prates, M. O., Lachos, V. H. & Cabral, C. R. B. (2012). mixsmsn: Fitting finite mixture of scale mixture of skew–normal distributions. *Journal of Statistical Software (A ser publicado)*.

[24] Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge.

[25] Steel, S. J. & Louw, N. (2001). Variable selection in discriminant analysis: measuring the influence of individual cases. *Computational Statistics and Data Analysis*, **37**, 249–260.

[26] Theodoridis, S. & Koutroumbas, K. (2008). *Pattern Recognition. Fourth Edition*. Academic Press, USA.

[27] Wang, H. & Hu, Z. (2009). On EM Estimation for Mixtures of Multivariate *t*–Distributions. *Neural Processing Letters*, **30**(3), 243–256.

[28] Wang, K., Ng, S. K. & McLachlan, G. J. (2009). Multivariate skew t mixture models: Applications to fluorescence-activated cell sorting data. *Proceedings of Digital Image Computing: Techniques and Applications*, pages 526–531.

[29] Xavier, C. M. (2012). *Segmentação, Classificação e Quantificação de Bacilos de Tuberculose em Imagens de Baciloscopia de Campo Claro Através do Emprego de uma Nova Técnica de Classificação de Pixels Utilizando Máquinas de Vetores de Suporte*. Dissertação de Mestrado – Universidade Federal do Amazonas, Amazonas.

[30] Zhu, J. & Hastie, T. (2004). Classification of Gene Microarrays by Penalized Logistics Regression. *Biostatistics*, **5**(3), 427–443.