UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

COLEGIADO DE ESTATÍSTICA

Mariana Machado Matheus

Aprendizado estatístico não supervisionado aplicado aos indicadores obstétricos dos municípios do Brasil

Vitória

2022

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

COLEGIADO DE ESTATÍSTICA

Mariana Machado Matheus

Aprendizado estatístico não supervisionado aplicado aos indicadores obstétricos dos municípios do Brasil

Monografia apresentada ao curso de graduação em Estatística do Departamento de Estatística, do Centro de Ciências Exatas da Universidade Federal do Espírito Santo, como requisito para obtenção do grau de Bacharel em Estatística.

Orientador(a): Agatha Sacramento Rodrigues
Coorientador(a): Bruno Ramos dos Santos

Vitória

2022

Aos meus irmãos.

Resumo

O aprendizado estatístico não supervisionado se refere a técnicas estatísticas de análise de dados não rotulados, de forma que não temos um gabarito para verificar a efetividade do método utilizado. Essas técnicas se fazem úteis quando se deseja criar ou avaliar a existência de subgrupos em relação a um conjunto de variáveis da população. Os métodos de agrupamento se baseiam em medidas de dissimilaridade, que quantificam a diferença entre observações, para realizar a partição dos dados, de forma que objetos contidos em um grupo sejam similares entre si e distintos dos demais. Técnicas mais tradicionais utilizam medidas de distância para essa partição, porém não é necessário que sejam limitadas a isso, em que métodos mais recentes trazem propostas diferentes de abordagem na temática de agrupamento. Tivemos como intuito, então, utilizar de diferentes técnicas de agrupamento para criar grupos de municípios brasileiros que se assemelham quanto a indicadores obstétricos obtidos através do Sistema de Informações sobre Nascidos Vivos (SINASC). Ao considerar a dimensão do Brasil, é natural que seus mais de 5.500 municípios apresentem características diversas em relação às mais variadas taxas, dentre elas, os indicadores obstétricos, que são importantes ferramentas para conhecimento e entendimento da qualidade da gestão da saúde materno-infantil de um município. Nesse trabalho foram discutidos diferentes métodos de agrupamento de dados, desde o clássico k-médias até alguns mais recentes, considerando outros métodos de particionamento, hierárquicos, de densidade e de agrupamento espectral. Foram discutidas, também, técnicas de validação e avaliação dos grupos formados a partir desses métodos. Alguns métodos apresentaram uma tendência a agrupar municípios com valores discrepantes, o que foi observado com mais intensidade nos hierárquicos aglomerativos, com exceção do método de Ward. Esse comportamento se refletiu nas medidas de avaliação, em que três das quatro métricas estudadas atribuíram a agrupamentos com esse problema os melhores índices. Foi utilizado, então, o índice de Calinski-Harabasz para selecionar o melhor agrupamento, que se mostrou mais apropriado aos dados do estudo, atribuindo melhores valores a agrupamentos que não apenas separavam *outliers*. Por esse índice o agrupamento feito pelo K-médias foi selecionado. Com o resultado do agrupamento, foi observado um grupo mais concentrado na região norte do país, que apresenta piores indicadores de acesso a serviço de saúde além de, analisando dados socioeconômicos do Censo brasileiro de 2010, ter apresentado piores indicadores de renda *per capita* e índice de desenvolvimento humano. Foi, ainda, realizado um estudo de simulação para melhor entendimento dos métodos estudados, avaliando a qualidade do agrupamento utilizando dados já rotulados, além de observar o comportamento das medidas de validação nesse cenário. Como resultado foi observado um comportamento que confirmava a adequação do K-médias utilizando como métricas de avaliação as medidas estudadas.

Palavras Chaves:

Análise de Agrupamento, Aprendizado Estatístico não Supervisionado, Indicadores Obstétricos, Saúde Materno-Infantil, SINASC, Validação de Agrupamento.

Abstract

The unsupervised statistical learning refers to statistical techniques of unlabeled data analysis, so that we do not have a template to verify the effectiveness of the method used. These techniques are useful when you want to create or evaluate the existence of subgroups in relation to a set of variables. The clustering methods are based on dissimilarity measures, which quantify the difference among the observations to perform the data partition, in a way that the objects contained in a group are similar to each other and distinct from others from other groups. Traditional techniques use distance measures for this partition, but it is not necessary that they be limited to this, as more recent methods bring different proposals for approaching the clustering thematic. We intended, then, to use different clustering techniques to create groups of Brazilian municipalities that are similar in terms of obstetric indicators obtained through the Information System on Live Births (SINASC). It is natural, given the size of Brazil, that its more than 5,500 municipalities have different characteristics in relation to the most varied rates, including obstetric indicators, which are important tools for knowledge and understanding of the quality of maternal and child health management in a municipality. In this work different clustering methods were discussed, from the classic K-means to more recent ones, considering other partitioning methods, hierarchical ones, density based and spectral clustering. Validation and evaluation techniques of the groups formed by these methods were also discussed. Some methods presented a tendency of grouping municipalities with discrepant values, what was observed with more intensity on the agglomerative hierarchical ones, with the exception of the Ward method. This behavior was reflected in the evaluation measures, in which three out of the four studied metrics attributed the best indexes to clusters with this problem. The Calinski-Harabasz index was then used to select the best clustering, which seemed to be more appropriate to the study data, giving better values to clusters that not only separated outliers. For this index the grouping made by K-means was selected. With the result of the clustering, a more concentrated cluster on the northern region of the country was observed, which presented worst health services access indicators, in addition to, analyzing socioeconomic data from the Brazilian Cesus of 2010, having presented worst *per capita* income and human development index. It was also performed a simulation study for better understanding of the studied methods, evaluating the quality of clustering by using already labeled data, in addition to observing the validation measures behavior in this scenario. As result was observed a behavior that confirmed the adequacy of the K-means using as evaluation metrics the studied measures.

Keywords:

Cluster Analysis, Unsupervised Statistical Learning, Obstetric Indicators, Maternal and Child Health, SINASC, Clustering Validation.

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

SINASC	Sistema de Informação sobre Nascidos Vivos
DN	Declaração de Nascido Vivo
ONU	Organização das Nações Unidas
IBGE	Instituto Brasileiro de Geografia e Estatística
PAM	Partition Around Medoids
CLARA	Clustering Large Applications
DIANA	Divisive Analysis
BIRCH	Balanced Iterative Reducing and Clustering using Hierarchies
CF	Clustering Feature
CF-tree	Clustering Feature-tree
ROCK	Robust Clustering using Links
DBSCAN	Density-based spatial clustering of applications with noise
OPTICS	Ordering Points To Identify the Clustering Structure
DB	Índice de Davies-Bouldin
D	Índice de Dunn
S	Índice da Silhueta
CH	Calinski-Harabasz
PCDaS	Plataforma de Ciência de Dados aplicada à Saúde
IDHM	Índice de Desenvolvimento Humano Municipal
UF	Unidade Federativa
PIB	Produto Interno Bruto
NJW	Algoritmo de Ng, Jordan e Weiss

Sumário

1 Introdução

O Brasil é um país de dimensões continentais, o que faz com que existam locais muito distintos entre si, tanto geograficamente quanto socialmente, e essas diferenças se manifestam, também, no acesso a saúde. Sendo esse um direito universal garantido pela Constituição Federal de 1988:

> Art. 196. A saúde é direito de todos e dever do Estado, garantido mediante políticas sociais e econômicas que visem à redução do risco de doença e de outros agravos e ao acesso universal e igualitário às ações e serviços para sua promoção, proteção e recuperação.

Dessa forma, todos têm direito a tratamentos adequados fornecidos pelo poder público de forma gratuita e igualitária, porém a desigualdade ao seu acesso é inegável e está associada não só a fatores econômicos como regionais, em que existem localidades no país onde a garantia a esse direito é dificultada.

No artigo XXV da Declaração Universal dos Direitos Humanos da ONU de 1948 é colocado que a maternidade e a infância têm direito à ajuda e assistência especiais. Nesse contexto de saúde de feto e gestante, é importante que se tenha conhecimento de índices que são capazes de informar a existência e a qualidade dessa assistência, como o acesso ao acompanhamento pré-natal, a situação da necessidade de cesárea, entre outros.

No Brasil, o Sistema de Informação sobre Nascidos Vivos (SINASC) é quem realiza a coleta de dados sobre os nascimentos informados em todo território nacional, tendo como documento padrão para essa coleta a Declaração de Nascido Vivo (DN). O SINASC vem, desde sua criação, fornecendo importantes dados sobre natalidade para todos os níveis do sistema de saúde, o que têm sua importância e qualidade comentadas em Jorge et al. (2007).

Dentre as informações fornecidas pelo SINASC estão as variáveis que indicam o município de residência da mãe, data do nascimento, peso e sexo do nascido, valores do apgar do primeiro e quinto minuto, anomalias congênitas, número de consultas de pré-natal realizadas pela gestante, idade gestacional no parto, tipo de gravidez (única, dupla, etc.), tipo de parto, entre outras.

Nesse sentido, se faz interessante entender como esses indicadores de saúde obstétrica se comportam por todo território, em especial em relação às menores unidades autônomas do país: os municípios.

É considerado município, pela Constituição de 1988, a unidade da federação com menor abrangência territorial, sendo, atualmente, contabilizados 5570 municípios pelo Instituto Brasileiro de Geografia e Estatística (IBGE) em todo território nacional, tendo a última atualização

sido realizada em 1º de janeiro de 2013, quando foram instalados mais 5 novos municípios. Esses estão distribuídos entre as 27 unidades federativas, que somam os 26 estados e o Distrito Federal, e podem ser acessados pelo portal do IBGE em https://cidades.ibge.gov.br.

Então, o agrupamento de localidades quanto a indicadores obstétricos se mostra útil para identificação de grupos semelhantes de regiões do país, para que, além de conhecer a distribuição do acesso e da qualidade a saúde no país, possam ser adotadas políticas de redução dessas desigualdades baseadas em dados.

O aprendizado estatístico não supervisionado (Xu e Tian, 2015) é a área da ciência capaz de realizar esses agrupamentos. A análise de agrupamento é conhecido como aprendizado não supervisionado porque as informações do *label* (rótulo) do grupo não estão presentes, o que difere do aprendizado supervisionado de classificação, uma vez que as informações do rótulo do grupo são fornecidas (Nasteski, 2017).

As técnicas de agrupamento procuram, por diferentes métodos, particionar os dados em grupos de forma que em um mesmo grupo estejam observações mais semelhantes entre si e objetos distintos sejam alocados em grupos separados. A análise de agrupamento é capaz, então, de identificar divisões nos dados até então desconhecidas, além de fornecer informações úteis a respeito do comportamento das variáveis nos dados, fazendo recortes que podem mostrar tendências interessantes a um estudo.

É importante, porém, salientar que a técnica de agrupamento é totalmente dependente das variáveis utilizadas na análise, assim, conclusões a respeito dos resultados obtidos devem ser feitas com cautela para que não se extrapole os limites de sua interpretação.

Por ser um assunto de grande interesse, existe uma variedade de estudos em análise de agrupamento, em que são propostas diferentes soluções para o desafio do aprendizado sem rótulos. Diferentes métodos realizam diferentes abordagens para identificar a similaridade entre as observações, fornecendo, então, diferentes resultados para o mesmo problema. Dessa forma, a ausência da real divisão existente dentro dos dados faz com que a validação de um agrupamento feito seja outro desafio, em que diferentes métodos de verificação da qualidade de um agrupamento são propostas, tendo em vista um cenário em que os reais grupos são mais densos e bem separados.

Essa monografia visa, então, utilizar os indicadores obstétricos do ano de 2019 dos municípios brasileiros, obtidos pelo SINASC, para realizar uma análise de agrupamento, utilizando diferentes métodos, a fim de identificar a presença de possíveis grupos de municípios que se assemelham em relação a esses indicadores. Faremos, ainda, uma avaliação do resultado obtido, observando as características dos grupos observados, tanto em relação às taxas obstétricas utilizadas quanto à indicadores socioeconômicos selecionados.

1.1 Objetivos

1.1.1 Objetivo geral

Realizar um agrupamento dos municípios brasileiros com relação aos indicadores obstétricos.

1.1.2 Objetivos específicos

- Identificar municípios *outliers*.
- Avaliar quais indicadores são mais importantes para o agrupamento realizado.
- Estudar a literatura de métodos de agrupamento, discutindo suas vantagens e desvantagens e cenários que são melhores empregados.
- Analisar os indicadores obstétricos em nível municipal, fazendo recortes daqueles municípios que apresentam os melhores e piores índices.

1.2 Organização

O presente trabalho será dividido da seguinte forma:

- Capítulo 2: **Metodologia**. Aqui serão abordados os métodos considerados neste trabalho, tais como: métodos de agrupamento por particionamento, por densidade, métodos hierárquicos de agrupamento e agrupamento espectral.
- Capítulo 3: **Aplicação**. Capítulo dedicado à aplicação dos métodos de agrupamento propostos aos dados de indicadores obstétricos, avaliação dos agrupamentos pelas medidas estudadas e análise dos resultados obtidos. Todas as análises foram feitas considerando a linguagem R (R Core Team, 2021).
- Capítulo 4: **Estudo de simulação**. Nesse capítulo será realizado um estudo de simulação para possibilitar maiores discussões a respeito dos métodos de agrupamento estudados e seus comportamentos em dados previamente rotulados, além de também possibilitar melhor entendimento das métricas de validação consideradas. Todo o estudo de simulação também foi realizado utilizando a linguagem R.
- Capítulo 5: **Conclusão**. Aqui serão apresentados os principais resultados obtidos pelo trabalho realizado.
- Capítulo 6: **Códigos**. Nesse capítulo consta os códigos utilizados para todo o desenvolvimento computacional realizado pelo estudo, considerando a linguagem R.

2 Metodologia

Neste capítulo será apresentado o embasamento teórico em que o trabalho foi fundamentado, com conceitos importantes apresentados na Seção 2.1. Os métodos de agrupamento considerados neste trabalho são apresentados ao longo da Seção 2.2 e as técnicas de validação dos resultados obtidos por esses estão na Seção 2.3.

2.1 Análise de Agrupamento

A análise de agrupamento tem como objetivo particionar o conjunto de dados em grupos, também chamados de *clusters*, de forma que indivíduos do mesmo grupo tenham características semelhantes entre si e distintas dos demais grupos. Assim, seja um conjunto de N observações de um vetor aleatório com p variáveis, cada observação será representada como um vetor $\mathbf{x}_i$ com p coordenadas, com $i = 1, \ldots, N$. A matriz de dados X é representada abaixo:

$$X = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_N \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1p} \\ x_{21} & x_{22} & \ldots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \ldots & x_{Np} \end{bmatrix}$$

Dessa forma, queremos localizar regiões no espaço vetorial de X que contém as maiores concentrações de observações, sem que tenhamos uma variável como referência para comparar o resultado obtido que nos indique a qualidade do agrupamento realizado. Por esse motivo, a análise de agrupamento é uma técnica não supervisionada.

Formalmente, os *clusters* C_l serão definidos por

- $C_l \neq \emptyset$, $l = 1, 2, \ldots, K$;
- $\bigcup_{l=1}^{K} C_l = \{1, 2, \cdots, N\}$;
- $C_j \bigcap C_l = \emptyset$, $j, l = 1, 2, \ldots, K$ e $j \neq l$.

Quando realizamos o agrupamento é importante que se utilize de um método que maximize as diferenças entre os *clusters* enquanto as minimiza dentro deles. Para isso surgem as medidas de similaridade, ou dissimilaridade, que quantificam a diferença entre as observações tomadas par a par.

As medidas de dissimilaridade mais utilizadas são a distância euclidiana e a distância euclidiana quadrática, respectivamente dadas por:

$$d(x_i, x_{i'}) = \sqrt{\sum_{j=1}^{p} (x_{ij} - x_{i'j})^2} \quad \text{e} \tag{1}$$

$$d^2(x_i, x_{i'}) = \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2.$$

Outras distâncias menos utilizadas são a distância absoluta (2) e a de Mahalanobis, que considera a matriz de covariância S, como pode ser visto na Equação (3).

$$d_a(x_i, x_{i'}) = \sum_{j=1}^{p} |x_{ij} - x_{i'j}| \tag{2}$$

$$d_M(x_i, x_{i'}) = \sqrt{(x_i - x_{i'})' S^{-1} (x_i - x_{i'})} \tag{3}$$

Uma forma comum de se representar as dissimilaridades entre os objetos de um conjunto de dados é através da matriz de dissimilaridade, definida como:

$$A = \begin{bmatrix} a(x_1, x_1) & a(x_1, x_2) & \dots & a(x_1, x_N) \\ a(x_2, x_1) & a(x_2, x_2) & \dots & a(x_2, x_N) \\ \vdots & \vdots & \ddots & \vdots \\ a(x_N, x_1) & a(x_N, x_2) & \dots & a(x_N, x_N) \end{bmatrix}, \tag{4}$$

em que $a(x_i, x_{i'})$ representa o valor da dissimilaridade entre x_i e $x_{i'}$, com $i,\ i' = 1, \dots, N$.

Dentre as utilidades da análise de agrupamento está a capacidade de identificar estratos da população em estudo, o que permite, também, que seja realizada uma amostragem considerando a existência dessas subpopulações para diminuição do conjunto de dados. As técnicas de agrupamento podem ser úteis, também, para a detecção de dados extremos, os *outliers*, que apresentarão comportamento mais distinto dos grupos formados.

Alguns conceitos são importantes para caracterizar os métodos de agrupamento, bem como os aspectos que podem ser usados para avaliar a utilização ou não de um determinado método de agrupamento. Essas características são:

- **Escalabilidade -** capacidade de lidar de forma eficiente com grandes bases de dados, uma vez que um grande banco de dados pode conter milhares, milhões ou até mesmo bilhões de observações.

- **Capacidade de lidar com diferentes tipos de variáveis -** muitos métodos são propostos para agrupar dados numéricos. No entanto, determinadas aplicações podem exigir o agrupamento de outros tipos de dados, como dados nominais e ordinais, ou combinações de mais de um tipo de dados.

- ***Clusters* com diferentes formatos -** muitos algoritmos de agrupamento determinam os *clusters* com base em medidas de distância e, por esse motivo, tendem a encontrar aglomerados esféricos com tamanho e densidade semelhantes. No entanto, um *cluster* pode ser de qualquer forma, como não esférico. É importante entender a capacidade dos algoritmos de detectar grupos de formas arbitrárias.

- **Capacidade de lidar com *outliers* -** a maioria dos conjuntos de dados do mundo real contém valores discrepantes. Os algoritmos de agrupamento podem ser sensíveis a esse ruído e podem produzir agrupamentos de baixa qualidade. Portanto, precisamos entender se determinado método de agrupamento é robusto ou não à *outliers*.

- **Capacidade de agrupar dados de alta dimensionalidade -** um conjunto de dados pode conter várias variáveis. A maioria dos métodos de agrupamento são bons para lidar com dados de baixa dimensão, como conjuntos de dados envolvendo apenas duas ou três dimensões. Encontrar *clusters* de objetos de dados em um o espaço dimensional é desafiador.

Existem variados métodos diferentes de agrupamento já descritos na literatura e técnicas mais avançadas vêm sendo estudadas e aprimoradas a fim de se contornar algumas limitações das técnicas já estabelecidas. É evidente que diferentes algoritmos podem fornecer diferentes agrupamentos para um mesmo conjunto de dados. No que segue são apresentados os métodos de agrupamento considerados nesse trabalho.

2.2 Métodos de agrupamento

Nessa seção serão tratados diferentes abordagens para a solução do problema de agrupamento. Os métodos mais tradicionais de agrupamento são os feitos por particionamento, que serão apresentados em 2.2.1, e os métodos hierárquicos, que serão abordados em 2.2.2.

Outros métodos podem realizar o agrupamento considerando a densidade de pontos no espaço, que tentam identificar as regiões mais densas do espaço vetorial separadas por regiões com menos objetos. Esses métodos são conhecidos como métodos baseados em densidade e nesse trabalho exploramos o DBSCAN na Seção 2.2.3.

Existe, ainda, uma classe de métodos que utiliza de técnicas de decomposição espectral para diminuição da dimensionalidade dos dados sem perda de informação dos grupos existentes. Esses são os chamados agrupamentos espectrais e serão abordados na Seção 2.2.4.

2.2.1 Métodos por particionamento

Ao considerar métodos por particionamento, cada partição representa um *cluster* com pelo menos uma observação. A maioria dos métodos de particionamento é baseada em distância.

Fixado previamente o número de grupos que serão criados, um método de particionamento cria uma separação inicial e usa uma técnica de realocação iterativa que tenta melhorar o particionamento movendo observações de um grupo para outro até que observações do mesmo grupo sejam próximas, enquanto que as observações em diferentes grupos estejam distantes.

O número de grupos a ser considerado costuma ser decidido por quem é o interessado no agrupamento a depender da necessidade e objetivo do estudo. Porém, a escolha desse parâmetro afeta diretamente a qualidade do resultado do agrupamento, por determinar a granularidade da análise.

Não somente nos métodos de particionamento, que precisam de um valor pré-estabelecido de grupos, a escolha do número de *clusters* deve ser feita com atenção, já que pode acabar levando à não identificação de grupos que deveriam existir ou à criação de subgrupos de um *cluster* que já estaria bem definido.

Uma forma simples de se definir o número de grupos é adotar $K = \sqrt{\frac{N}{2}}$, porém esse método se torna menos efetivo para grandes conjuntos de dados já que é diretamente proporcional ao número N de observações.

Um método mais comum é o chamado método do cotovelo, que considera a relação entre a variância total dentro dos *clusters* e o número de grupos criados. Ele considera que o aumento do valor de K diminuirá a variância, porém, em determinado ponto, esse aumento já não irá oferecer uma melhoria significativa na granularidade do agrupamento. Esse valor de K seria o ponto ótimo que pode ser visto tracejado na Figura 1.

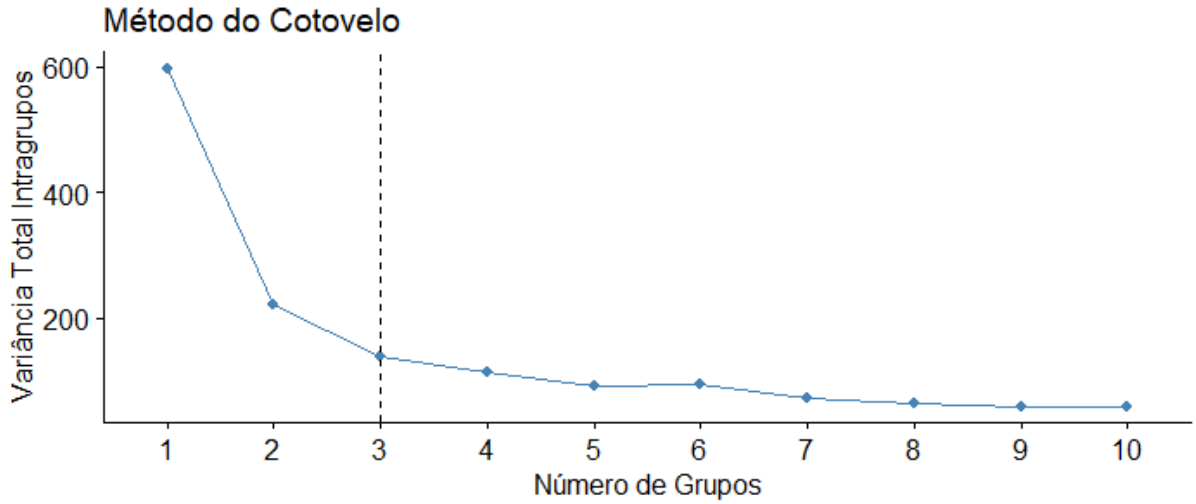


Figura 1 – Representação do método do cotovelo de seleção no número de *clusters*.

A variância total intragrupo mais comum é obtida através das distâncias euclidianas quadráticas entre as observações e o centróide daquele mesmo grupo. Os centróides c_l são representantes dos grupos C_l formados, sendo obtido pelo ponto médio entre as observações

atribuídas àquele *cluster*, como apresentado abaixo:

$$c_l = \frac{1}{|C_l|} \sum_{i \in C_l} x_i. \tag{5}$$

Assim, temos que a variância total intragrupo é dada pela equação

$$\sum_{l=1}^{K} \sum_{i \in C_l} d^2(x_i, c_l). \tag{6}$$

A seguir são apresentados os métodos por particionamento considerados nesse trabalho: k-médias e k-medóides com os algoritmos PAM e CLARA.

2.2.1.1 K-médias

Um dos mais famosos métodos quando se trata de agrupamento, o K-médias se caracteriza por buscar encontrar as K partições do dado de forma a minimizar a variância dada pela Equação (6).

O resultado obtido por esse método é dependente dos centróides iniciais, não sendo possível, assim, minimizar a variância de forma exata. Temos, então, o algoritmo proposto por Lloyd (1982) abaixo para o agrupamento pelo método k-médias:

1. Escolha dos K centróides iniciais, que podem ser observações escolhidas aleatoriamente, determinadas observações ou coordenadas definidas pelo pesquisador;
2. Particionar os dados de forma que cada observação esteja no C_l cuja distância para o c_l seja a menor;
3. O centróide de cada *cluster* é atualizado com as novas observações atribuídas a ele no passo 2;
4. Repetir os passos 2 e 3 até que não tenha mais mudança de agrupamento.

É possível, também, determinar um número máximo de iterações a serem realizadas pelo método, o que melhora seu custo computacional e o torna viável para conjunto de dados mais extensos.

O algoritmo de Hartigan-Wong (Hartigan e Wong, 1979) é uma alternativa que considera uma etapa de validação para mudança do agrupamento feito. Nele é verificado, a cada iteração do código, se o centróide de algum grupo foi atualizado em relação à última. Nesse caso, esse novo objeto só será atribuído ao *cluster* se sua soma das distâncias quadráticas for diminuída.

O método K-médias apresenta limitações ao lidar com *clusters* com formas não convexas ou grupos de tamanhos muito diferentes. Além disso, esse método se torna sensível a dados extremos, os *outliers*, já que, ao ser adicionado a um grupo, pode influenciar substancialmente no valor do novo centróide.

2.2.1.2 K-medóides

Em situações em que existem observações com valores extremos, o valor central representante de cada grupo pode sofrer forte influência caso seja obtido pelo cálculo do centróide.

Para isso surgem os algoritmos K-medóides que resolve o problema de agrupamento minimizando a variância total intragrupo, assim como K-médias. Porém, ao invés de utilizar o valor médio das observações do grupo para obter esse ponto, eles consideram uma observação dos dados contida no grupo como o ponto referencial central.

O algoritmo PAM (*Partitioning Around Medoids*) proposto por (Kaufman e Rousseeuw, 1990, Cap. 2) considera um custo para os reagrupamentos feitos a cada iteração do algoritmo, que se dá da seguinte forma:

1. escolha dos K medóides iniciais;
2. as observações não selecionadas como medóides são atribuídas ao grupo cujo medóide é o mais próximo;
3. selecionar aleatoriamente uma observação não medóide o_r;
4. calcular o custo de se mudar o medóide atual para o_r;
5. caso o custo seja menor que 0, então a troca de medóide é feita;
6. voltar ao passo 2 até que não tenha mais mudança de agrupamento.

O custo de mudança do medóide atual para outra observação é a diferença da variância total intragrupo calculada considerando o_r como representante e considerando o então medóide. Essa variância tem, então, um formato diferente da disposta na Equação (6), em que ao invés do centróide c_l, é considerada a distância para um medóide. Além disso, é comum se considerar a medida de distância absoluta (2) no lugar da distância euclidiana quadrática.

Como o algoritmo PAM realiza trocas de medóide repetidas vezes, isso pode ter um maior custo computacional à medida que aumentamos o número de observações. Então, para viabilizar a implementação do K-medóides em grandes massas de dados, surge o algoritmo CLARA (*Clustering Large Applications*), proposto também por Kaufman e Rousseeuw (1990)[Cap. 3], que seleciona uma amostra de observações e realiza o PAM nesses dados de menor tamanho. Dessa forma, seu algoritmo está descrito abaixo:

1. dividir o conjunto de dados em subconjuntos (amostras);
2. computar o algoritmo PAM em cada amostra e retornar os medóides identificados;

3. para cada conjunto de medóides observado em cada amostra, os dados são particionados de forma que cada observação não medóide seja atribuída ao *cluster* do medóide mais próximo;

4. a variância total intragrupo é calculada para cada diferente agrupamento gerado;

5. a partição que apresentar menor variância total intragrupo é a selecionada.

2.2.2 Métodos Hierárquicos

No método hierárquico os agrupamentos são realizados em diferentes níveis, em que no maior dos níveis todos os dados estão em um grupo apenas, enquanto no mais baixo nível cada *cluster* é composto por uma única observação, o que o difere dos métodos por particionamento, que tem sua quantidade de grupos fixada.

Os agrupamentos podem ser feitos através de duas abordagens, são elas a **aglomerativa** e a **divisiva**, ambas possuindo $N - 1$ níveis hierárquicos, em que a primeira inicia o processo no nível mais baixo e a cada etapa pares de grupos são fundidos formando um, até que se atinja o nível hierárquico mais alto com um *cluster*. Assim, a cada nova iteração temos um grupo a menos, em que o par de grupos que será unido é eleito baseado nos que apresentam a menor dissimilaridade. Em contrapartida, a abordagem divisiva inicia no nível mais alto da hierarquia e divide um grupo em dois novos *clusters* a cada iteração, divisão essa que é feita de forma a produzir dois grupos com a maior dissimilaridade entre eles.

Para ser realizado, o método hierárquico aglomerativo se baseia na medida de dissimilaridade entre dois grupos, que é obtida pela distância entre eles, para qual comumente é utilizada a distância euclidiana. Na literatura de métodos de agrupamento essa distância também é chamada de *linkages* e podem ser definidas da seguinte forma:

- Método do vizinho mais próximo (*Single linkages*): considera a menor distância entre todos os possíveis pares de observação entre dois grupos, dada por:

$$d(C_l, C_{l'}) = \min_{x_i \in C_l; x_j \in C_{l'}} d(x_i, x_j). \tag{7}$$

- Método do vizinho mais distante (*Complete linkages*): utiliza a maior distância entre todos os possíveis pares de observação entre dois grupos, dada por:

$$d(C_l, C_{l'}) = \max_{x_i \in C_l; x_j \in C_{l'}} d(x_i, x_j).$$

- Método da média das distâncias (*Average linkages*): considera a média das distâncias entre os pares de observação entre dois grupos, dada por:

$$d(C_l, C_{l'}) = \frac{1}{|C_l||C_{l'}|} \sum_{x_i \in C_l; x_j \in C_{l'}} d(x_i, x_j).$$

- Método do centróide (Centróide *linkages*): considera a distância entre os centróides descritos na Equação (5) de cada grupo como a dissimilaridade entre eles, dada por:

$$d(C_l, C_{l'}) = d^2(c_l, c_{l'}).$$

- Método de Ward: introduzido por Ward (1963), ele busca, a cada iteração, fundir os grupos que minimizam a variância dentro do novo grupo formado, de forma similar ao método K-médias. A distância é dada por:

$$d(C_l, C_{l'}) = \frac{n_l n_{l'}}{n_l + n_{l'}} d^2(c_l, c_{l'}).$$

Quando nos referimos a métodos hierárquicos de agrupamento, a abordagem aglomerativa é a mais difundida e trabalhada. Isso acontece pelo fato do custo computacional da abordagem divisiva ser mais alto, já que é necessário, a cada iteração, identificar a melhor divisão do grupo para que sua dissimilaridade seja máxima. Assim, o algoritmo para o agrupamento pelo método hierárquico aglomerativo será:

1. cada uma das n observações será atribuída a um grupo C_l;
2. a partir do *linkage* selecionado, é calculada a medida de dissimilaridade entre os pares de grupos;
3. o par de *clusters* que apresentar o menor valor de dissimilaridade é fundido em um só;
4. repetir os passos 2 e 3 até que todas as observações estejam em um grupo.

Um algoritmo para agrupamento hierárquico divisivo é o Divisive Analysis (DIANA) (Kaufman e Rousseeuw, 1990, Cap. 6), que seleciona o grupo com maior dissimilaridade entre um de seus pares de observações para realizar a divisão e separa o objeto que possui maior dissimilaridade média com os demais. Assim, são identificadas as observações que são mais similares a essa e dissimilar às demais e divide o grupo. O algoritmo para o agrupamento DIANA é dado por:

1. todas as observações são agrupadas em um grande grupo;
2. a observação com maior distância média para os pontos do mesmo grupo é separada em um grupo;
3. cada observação do grupo inicial será atribuída ao novo grupo caso a distância média para seus objetos seja menor que a distância média para os demais pontos do grupo inicial;
4. calcula-se o diâmetro de todos os grupos, que é a maior distância entre duas de suas observações, e selecionamos o que apresentar o maior valor;

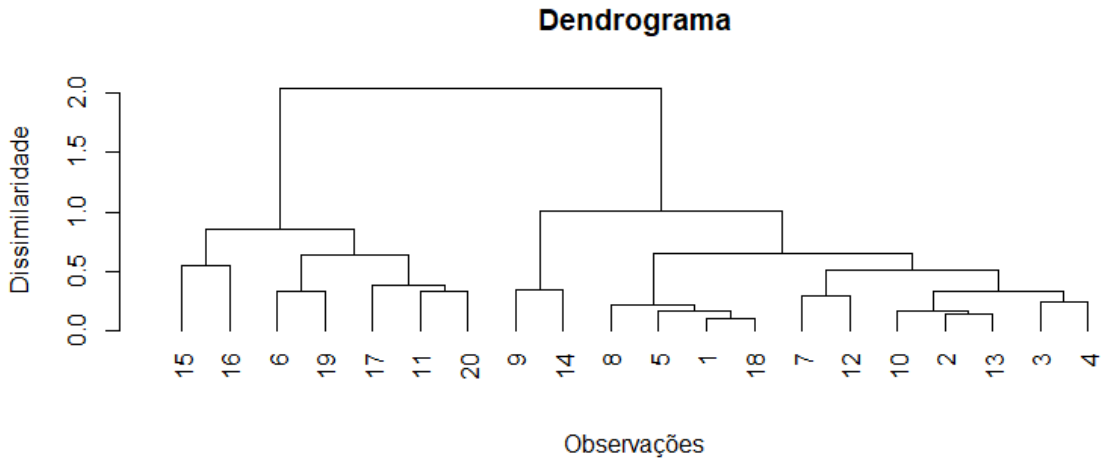


Figura 2 – Representação de um dendrograma.

5. voltar ao passo 2 até que todas as observações sejam as únicas de seu grupo.

Para a visualização dos *clusters* formados no agrupamento hierárquico existe o dendrograma, que é um gráfico ramificado em que as ramificações representam as junções ou divisões dos *clusters* a cada etapa. A altura do ramo para seu primeiro nó quantifica a dissimilaridade entre os grupos divididos. Uma representação de um dendrograma pode ser vista na Figura 2.

Como no problema do agrupamento queremos separar observações diferentes entre si, é razoável que os grupos apresentem maiores valores de dissimilaridade, o que, no dendrograma é representado pela altura do ramo. Assim, para determinar o número de grupos a partir do dendrograma, procuramos por uma grande diferença de altura (dissimilaridade) entre o acréscimo de um *cluster* aos dados.

Uma característica dos métodos hierárquicos apresentados até agora é o fato de que um agrupamento ou divisão feito não é desfeito, ou seja, não é realizada a troca de observações entre os *clusters*. Assim, decisões de união ou divisão que não forem bem feitas, podem levar a grupos de baixa qualidade. Além disso, os métodos não são bem dimensionados porque cada decisão de mesclagem ou divisão precisa examinar e avaliar muitos objetos ou *clusters*.

2.2.2.1 BIRCH

Um problema comum entre os métodos de *clustering* apresentados é o custo computacional elevado devido ao grande número de iterações necessárias para se realizar o agrupamento dos dados. Para isso foi proposto o método de agrupamento BIRCH (*Balanced Iterative Reducing and Clustering using Hierarchies*) (Zhang et al., 1996). Esse método considera a limitação de espaço e memória para o problema de agrupar observações de um conjunto de dados através da realização da partição em etapas, o que o torna eficiente para grandes massas de dados.

Além disso, o método BIRCH supera a incapacidade de desfazer o que foi feito no passo anterior dos métodos aglomerativos comentados anteriormente.

Para realização do agrupamento pelo BIRCH são utilizados os conceitos de *Clustering Feature* (CF) e CF-*tree*. O CF é um vetor com a informação que o algoritmo irá carregar a respeito de cada grupo, em que, dado um *cluster* C_l, seu *clustering feature* será dado por $CF_l = (n_l, LS_l, SS_l)$, em que n_l é o número de objetos contido em C_l, LS_l é a soma linear das observações do grupo e SS_l é a soma quadrática dessas observações, ou seja,

$$LS_l = \sum_{i \in C_l} x_i \qquad \text{e} \qquad SS_l = \sum_{i \in C_l} x_i^2.$$

Essa característica de carregar apenas um sumário de tamanho fixo de cada *cluster* que faz dele um método eficiente em termos de custo computacional (não é necessário trabalhar com os dados todo o tempo) e sem perda de informação, já que é possível facilmente obter muitas estatísticas de um grupo a partir de seu CF, como os centróides de cada grupo, a distância média R_l dos pontos de um *cluster* para seu centróide e a distância média D_l par a par entre seus objetos, dados respectivamente pela Equação (5) e por:

$$\begin{aligned} R_l &= \left(\frac{\sum_{i \in C_l} (x_i - c_l)^2}{n_l} \right)^{1/2} = \left(\frac{n_l SS_l - 2LS_l^2 + n_l LS_l +}{n_l^2} \right)^{1/2} \qquad \text{e} \\ D_l &= \left(\frac{\sum_{i \in C_l} \sum_{i' \in C_l} (x_i - x_{i'})^2}{n_l (n_l - 1)} \right)^{1/2} = \left(\frac{2n_l SS_l - 2LS_l^2}{n_l (n_l - 1)} \right)^{1/2}. \end{aligned} \tag{8}$$

Além disso, os CFs são aditivos. A partir da informação contida no vetor CF temos que a junção de dois grupos disjuntos será dada pela soma de seus CFs, dada por:

$$CF_l + CF_{l'} = (n_l + n_{l'}, LS_l + LS_{l'}, SS_l + SS_{l'}).$$

Já a CF-*tree* é uma árvore que reúne os CFs dos *clusters*. Uma representação dela pode ser vista na Figura 3. Ela possui o parâmetro de fator de ramificação B que determina o máximo de subgrupos, chamados de "filhos", de cada nó não-folha, que, por sua vez, estocam as somas de CFs de seus filhos, sumarizando a informação de clusterização contida neles.

Os nós da CF-*tree* também possuem subgrupos que compõem um novo *cluster* formado, porém todos esses subgrupos devem ter diâmetro D_l ou raio R_l (Equação (8)) menor que um limiar T pré-estabelecido.

O algoritmo BIRCH tenta produzir os melhores *clusters* com os recursos disponíveis. Para isso, destacamos duas fases:

- **Fase 1:** o algoritmo lê o conjunto de dados e constrói uma CF-*tree* inicial, a qual pode ser vista como uma compressão multinível dos dados.

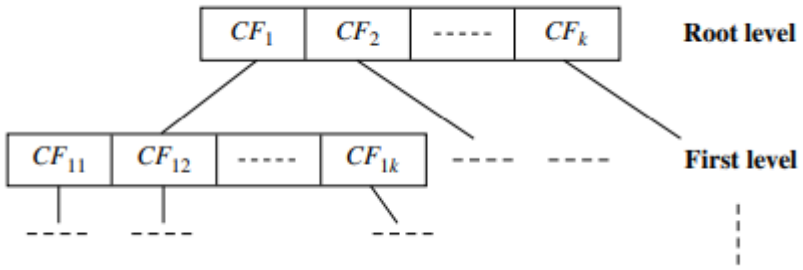


Figura 3 – Representação de uma CF-*tree* do método de agrupamento BIRCH.

Fonte: Han et al. (2012)

- **Fase 2:** o algoritmo aplica um segundo algoritmo para clusterizar os nós folhas da CF-*tree*, o qual remove *clusters* esparsos como outliers e agrupa *clusters* densos dentro de outros ainda maiores.

Na fase 1, a CF-*tree* é construída dinamicamente à medida que os objetos são inseridos de maneira incremental. Um objeto é inserido na folha mais próxima (*subcluster*). Se o diâmetro do *subcluster* armazenado no nó folha após a inserção for maior que o valor limiar T, o nó folha deve ser criado. Após a inclusão de um novo objeto, a informação sobre ele deve ser propagada até a raiz da árvore.

O tamanho da CF-*tree* pode ser alterado modificando o limiar T, construindo uma nova árvore e reorganizando os CFs nas folhas. Com T maior, um CF pode receber mais dados e grupos poderão se juntar e a árvore ficará menor. Dessa maneira, a árvore CF é reconstruída. O processo de reconstrução é realizado construindo uma nova árvore a partir dos nós folha da árvore antiga. Assim, o processo de reconstrução da árvore é feito sem a necessidade de reler todas as observações. Uma vez que a árvore CF é construída, qualquer algoritmo de agrupamento, como K-médias, pode ser usado com a árvore CF na Fase 2.

2.2.2.2 Outros métodos hierárquicos

O método de agrupamento Chameleon apresentado em Karypis et al. (1999) também utiliza processos hierárquicos de agrupamento, fazendo isso em etapas. Na primeira etapa, é realizada a partição dos dados, utilizando um método de partição por grafos, no caso os grafos dos vizinhos mais próximos. Na segunda etapa, os grupos formados são agrupados através de um algoritmo aglomerativo.

Quando lidamos com variáveis numéricas, a utilização de cálculos de distância para medir a dissimilaridade entre os objetos do conjunto de dados se mostra apropriada. No entanto, quando contamos com variáveis categóricas, essas métricas podem não ser válidas. Uma abordagem mais utilizada no caso de variáveis binárias é a utilização do coeficiente de Jaccard, apresentado na

Equação (9), que, por sua vez, apresenta algumas limitações que são demonstradas por Guha et al. (2000) e aumentam progressivamente a medida que aumentamos a dimensão dos dados.

$$J(x_i, x_{i'}) = \frac{|x_i \cap x_{i'}|}{|x_i \cup x_{i'}|} \tag{9}$$

Assim, surge o agrupamento ROCK (*Robust Clustering using Links*), também proposto em Guha et al. (2000), um método hierárquico aglomerativo, que, no lugar da distância Euclidiana ou do coeficiente de Jaccard, utiliza *links* entre os objetos, que considera como similaridade o número de observações próximas aos dois objetos.

2.2.3 DBSCAN

Os métodos por particionamento e de agrupamento hierárquico são adequados para encontrar agrupamentos esféricos ou convexos, ou seja, funcionam bem para clusters compactos. Então, para encontrar *clusters* com formas arbitrárias, foi proposto o método DBSCAN (Ester et al., 1996), que tenta identificar as regiões mais densas do espaço vetorial separadas por regiões com menos objetos. A ideia geral desse algoritmo é a de identificar os *clusters* de forma que a densidade de pontos ao redor de cada ponto de um grupo seja maior que um limite estabelecido.

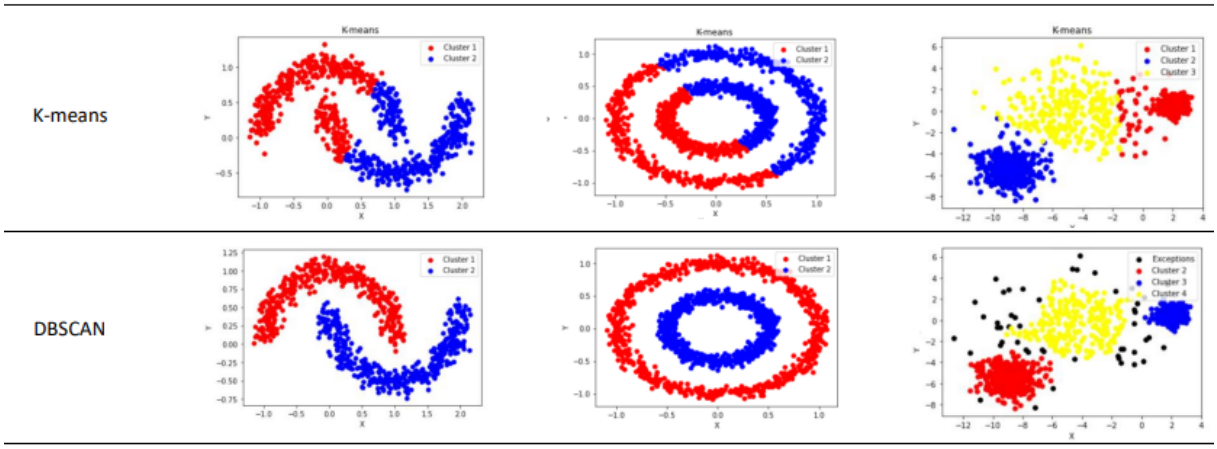


Figura 4 – Comparação de funcionamento dos métodos DBSCAN e K-médias (K-*means*).

Fonte: Boyko et al. (2021)

A Figura 4 mostra a diferença do funcionamento do método de particionamento K-médias e do DBSCAN em conjuntos de dados com grupos não convexos e de diferentes formatos. Nele podemos ver, além da eficiência da detecção correta dos grupos nos dois primeiros conjuntos de dados, que ele foi capaz, também, de identificar os *clusters* mais esféricos do último exemplo e localizar pontos que não são pertencentes a nenhum deles, os *outliers*.

Para o entendimento do agrupamento DBSCAN são definidos os seguintes termos:

- ϵ-vizinhos: os ϵ-vizinhos de um ponto x_i são os objetos cuja distância para x_i seja menor ou igual a ϵ, em que ϵ é um parâmetro de raio pré definido. Assim, a vizinhança $NEps$ de um ponto x_i será dada por $NEps(x_i) = \{x_j \in D | dist(x_i, x_j) \leq \epsilon\}$, em que D é o conjunto de dados.

- Ponto de Núcleo: um ponto x_i é um ponto de núcleo se seu número de ϵ-vizinhos for maior ou igual a um valor mínimo estabelecido, chamado de MinPts, de forma que $|NEps(x_i)| \geq MinPts$.

- Ponto de Borda: um ponto x_i é um ponto de borda se seu número de ϵ-vizinhos for menor ao MinPts estabelecido, ou seja $|NEps(x_i)| < MinPts$, mas algum ponto de núcleo x_j está em sua vizinhança, de forma que $x_j \in NEps(x_i)$.

- Diretamente alcançável por densidade: um ponto x_j é dito diretamente alcançável por densidade por um ponto x_i se x_j é um ϵ-vizinho de x_i, isso é, $x_j \in NEps(x_i)$, e se x_i for ponto de núcleo.

- Alcançável por densidade: um ponto x_i é dito alcançável por densidade por um ponto x_j se existe uma cadeia de pontos $x_{i'}$, com $i' \leq N \in \mathbb{N}$, em que $x_1 = x_j$, $x_N = x_i$ e $x_{i'+1}$ é diretamente alcançável por densidade por $x_{i'}$.

- Conectado por densidade: um ponto x_i é dito conectado por densidade a um ponto x_j se existir um terceiro ponto $x_{i'}$ em que ambos são alcançáveis por densidade por $x_{i'}$.

Temos, então, que os grupos pelo método de agrupamento DBSCAN serão definidos a partir dos critérios a seguir:

- Seja $x_i \in C_l$, se x_j é alcançável por densidade por x_i, então $x_j \in C_l$.

- Todos os pontos de um *cluster* são conectados por densidade entre si.

A partir dos critérios estabelecidos para que os pontos sejam agrupados em *clusters*, o método DBSCAN é capaz de identificar *outliers* no conjunto de dados, já que, diferente de outros métodos, ele não aloca todos os objetos em grupos. Dessa maneira, os *outliers* são definidos como os pontos que não são atribuídos a nenhum *cluster*.

Para o algoritmo de agrupamento via método DBSCAN, primeiramente todas as observações da base de dados são classificadas como "não visitadas" e então os próximos passos são seguidos:

1. Selecionar aleatoriamente uma observação "não visitada" x_i;

2. Verificar o número de ϵ-vizinhos da observação x_i selecionada;

a) caso $|NEps(x_i)| \geq MinPts$, um novo *cluster* C_l é criado para x_i, e todos os objetos na ϵ-vizinhança de x_i são adicionados a um conjunto candidato, M.

   O algoritmo adiciona iterativamente a C_l todos objetos em M que não pertencem a nenhum *cluster*.

   Nesse processo, para um objeto x_j em M que carrega o rótulo "não visitado", o algoritmo o marca como "visitado" e verifica sua ϵ-vizinhança.

   Se $|NEps(x_j) \geq MinPts$, aqueles objetos na ϵ-vizinhança de x_j são adicionados a M.

   DBSCAN continua adicionando objetos a C_l até que M esteja vazio. Neste momento, o *cluster* C_l é concluído.

b) caso $|NEps(x_i)| < MinPts$, x_i é marcado como um ponto de ruído;

3. Repetir os passos 1 e 2 até que não tenha observação "não visitada".

O DBSCAN possui, então, dois parâmetros que devem ser previamente estabelecidos, sendo eles o ϵ e o MinPts. A especificação desses valores afeta diretamente o resultado do agrupamento, já que determina o tamanho mínimo de cada grupo e o quão distante cada *cluster* será de outro. O fato desses dois parâmetros serem globais, ou seja, definidos para todos os grupos a serem formados é a principal limitação desse algoritmo, já que diferentes *clusters* podem apresentar diferentes densidades de pontos.

Para a seleção do valor de MinPts é comum que seja baseado no problema em si ou conhecimento prévio. Porém, uma regra proposta por Sander et al. (2004) diz que o MinPts = $2 \times p$ pode ser um bom valor para o parâmetro. Já o ϵ costuma ser baseado no número mínimo de pontos definido, em que é construído o gráfico das distâncias de cada ponto para os MinPts-1 vizinhos mais próximos e escolhido o valor "cotovelo", ou seja, onde começa a ocorrer um crescimento mais acentuado nas distâncias ordenadas.

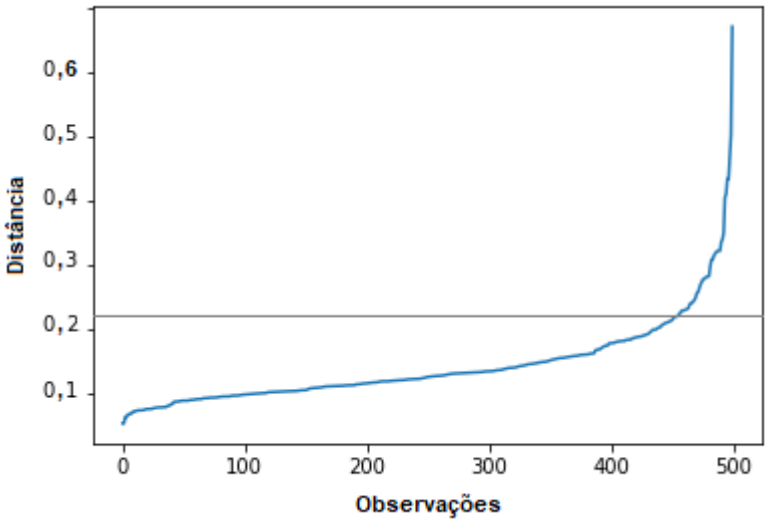


Figura 5 – Representação de um gráfico de distâncias para os vizinhos mais próximos.

Uma representação pode ser vista na Figura 5, em que no eixo x estão as N observações em ordem crescente em relação ao valor da sua respectiva distância para os MinPts-1 vizinhos mais próximos, no eixo y estão essas distâncias e a linha que corta o gráfico mostra onde seria o ϵ apropriado.

Como solução para a limitação dos parâmetros globais do DBSCAN, foi proposto o algoritmo OPTICS (Ankerst et al., 1999), que, de maneira geral, ordena os dados de forma que observações que estão em grupos mais densos estarão mais próximos nessa ordenação. Mais informações sobre esse método pode ser consultadas em Han et al. (2012).

2.2.4 Agrupamento Espectral

Alguns métodos de agrupamento apresentam problemas para lidar com conjuntos de dados com maiores dimensões, ou seja, que contam com um número grande de variáveis, podendo exigir um custo computacional maior. Os métodos de agrupamento espectral surgem como uma abordagem que conta com redução de dimensionalidade sem perda de informação contida nas variáveis, podendo acentuar a distância entre grupos distintos que, por conta da alta dimensionalidade dos dados, não estavam tão bem separados.

Os algoritmos de agrupamento espectral utilizam de uma medida de dissimilaridade para a representação por grafos do conjunto de dados, em que temos $G(V, A)$ com o conjunto de vértices $V = v_1, ..., v_N$ equivale ao conjunto de observações e as arestas A correspondem a uma matriz de similaridade, apresentada na Equação (4), em que $a(x_i, x_{i'})$ será o peso da aresta que incide sobre x_i e $x_{i'}$. Então, temos a solução para o agrupamento em K *clusters* como um problema de corte de arestas, que, como discutido em Filippone et al. (2008), apresenta alta complexidade, o que traz como alternativa a utilização da teoria espectral para o problema.

Definimos uma matriz diagonal D que contém os graus de cada vértice em sua diagonal, obtidos pela soma abaixo:

$$d_{ii} = \sum_{i'=1}^{N} a_{ii'}.$$

Construímos, então, uma Matriz Laplaciana que pode variar a depender do algoritmo utilizado. Nesse trabalho seguiremos com o algoritmo de Ng et al. (2001), que utiliza uma normalização de A. Porém, outras abordagens podem ser realizadas, como visto em Verma e Meila (2003).

Assim, segue o algoritmo de Ng, Jordan e Weiss (NJW), que tem como parâmetro pré-estabelecido, assim como o K-médias (Seção 2.2.1.1), o número K de grupos a ser formado:

1. Calculamos a matriz de similaridade A, que conta com um parâmetro escalar σ, definida por:
$$a_{ii'} = \begin{cases} exp\left(-\frac{||x_i - x_{i'}||^2}{2\sigma^2}\right), & i \neq i' \\ 0, & i = i' \end{cases};$$
2. Obtemos a matriz de graus D;
3. Obtemos a Matriz Laplaciana L, em que
$$L = D^{-1/2} A D^{-1/2};$$
4. Identificamos os K maiores autovalores de L e armazenamos em colunas seu autovetores associados em uma matriz $Z_{N \times K}$;
5. Definimos uma nova matriz Y normalizada, de forma que
$$y_{ii'} = \frac{z_{ii'}}{\sqrt{\sum_{i'} z_{ii'}^2}};$$
6. Com a redução de dimensão dos dados em uma matriz $N \times K$, realizamos algum agrupamento nos dados de Y, como o K-médias.

A seleção dos parâmetros K e σ pode, assim como nos demais métodos discutidos até aqui, ser feita de forma arbitrária a depender do objetivo do estudo. Existem, porém, diferentes abordagens que podem ser tomadas, como pode ser visto em Zelnik-manor e Perona (2004).

Nesse trabalho optamos por fixar o parâmetro $\sigma = 1$ e definimos o número K de acordo com os valores dos primeiros autovalores da Matriz Laplaciana calculada. Dessa forma, o K ótimo para o estudo será o que, após ele, ocorre queda no autovalor de L, visto que os maiores autovalores dessa matriz estarão próximos de 1, como exemplificado na Figura 6.

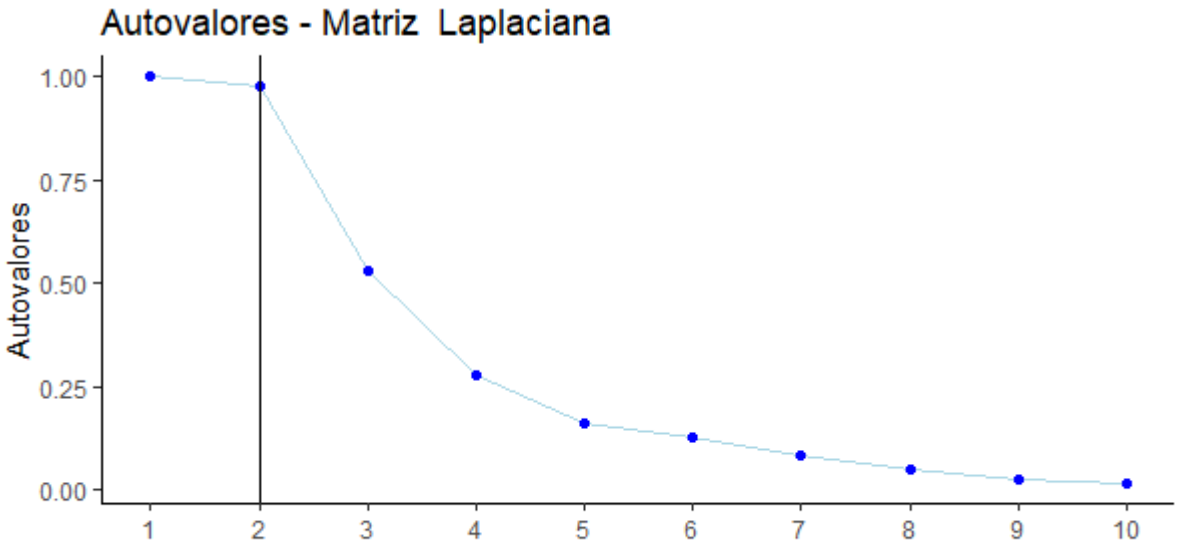


Figura 6 – Representação da seleção de K para o agrupamento espectral.

2.3 Validação

Como discutido em Halkidi et al. (2001), no problema do agrupamento não temos como verificar o grau de acerto do resultado obtido, já que os verdadeiros grupos não são conhecidos a *priori*. Dessa forma, algum tipo de validação deve ser aplicado à partição final.

Os chamados critérios de validação internos são medidas de avaliação de qualidade do agrupamento que considera apenas as variáveis disponíveis nos dados, que não contam com o real agrupamento das observações. Nesse trabalho foram selecionados 4 diferentes índices de avaliação interna da qualidade do particionamento realizado, são eles: Davies-Bouldin, Dunn, Silhueta e Calinski-Harabasz.

Outros métodos de validação internos e discussão sobre a qualidade de cada um podem ser encontrados em Liu et al. (2010). Os métodos de validação considerados nesse trabalho são melhores descritos no que segue.

1. **Davies-Bouldin (DB)**

   Proposto por Davies e Bouldin (1979), o índice tem como ideia geral fornecer a média da similaridade entre cada grupo e seu grupo mais similar dentre os demais *clusters* obtidos. Dessa forma, para cada grupo l definido, calculamos uma distância média δ_l de suas observações a um valor referencial m_l, podendo ser um centróide ou medóide. Assim, temos que δ_l será dada por

$$\delta_l = \left(\frac{1}{n_l} \sum_{i=1}^{n_l} ||x_i - m_l||^q \right)^{\frac{1}{q}},$$

em que $q \in \mathbb{N}$ é pré definido, sendo seus valores mais comumente usados são $q = 1$, que corresponde à média das distâncias absolutas, apresentada na Equação (2), e $q = 2$, que seria o equivalente ao cálculo do desvio padrão intragrupo.

Para esse índice é utilizada, também, a distância entre os grupos $\Delta_{ll'}$, que é obtida como a distância entre os valores referenciais de cada grupo:

$$\Delta_{ll'} = \left(\sum_{j=1}^{p} |m_{jl} - m_{jl'}|^t \right)^{\frac{1}{t}},$$

em que $t \in \mathbb{N}$ também é pré definido e costuma ser adotado como $t = 1$, que corresponde à distâncias absolutas e $t = 2$, que seria a distânca euclidiana, apresentada na Equação (1). Assim, o índice **DB** de Davies-Bouldin fica definido por:

$$\mathbf{DB} = \frac{1}{K} \sum_{l=1}^{K} \max_{l \neq l'} \left(\frac{\delta_l + \delta_{l'}}{\Delta_{ll'}} \right).$$

Como procuramos agrupar observações de forma a minimizar a variância intragrupo e maximizar a diferença entre eles, conclui-se que valores **menores** do índice Davies-Bouldin são melhores.

2. **Dunn (D)**

   Proposto por Dunn (1974), o índice de Dunn mede a razão entre a separação dos grupos e a variância dentro deles, dessa forma, **maiores** valores são melhores.

   O cálculo desse índice considera a separação entre dois grupos $d(C_l, C_{l'})$ pela distância do vizinho mais próximo, apresentada na Equação (7). Já a variância intragrupo considerada é o que seria o diâmetro $diam_l$ desse *cluster*, obtido por

   $$diam_l = \max_{i \neq i'; i,i' \in C_l} d(x_i, x_{i'}).$$

   Assim, temos que o índice **D** de Dunn será da forma

   $$\mathbf{D} = \frac{\min_{l,l' \in \{1,\ldots,K\}; l \neq l'} d(C_l, C_{l'})}{\max_{l \in \{1,\ldots,K\}} diam_l}.$$

3. **Silhueta (S)**

   O índice da Silhueta **S** mede a qualidade do agrupamento considerando as distâncias de cada ponto às observações do mesmo grupo e aos demais *clusters* formados. Para isso definimos para a i-ésima observação, pertencente ao grupo C_l, as medidas a_i e b_i abaixo:

   $$\begin{aligned} a_i &= \frac{1}{n_l - 1} \sum_{i \neq i'; i,i' \in C_l}^{n_l} d(x_i, x_{i'}) \\ b_i &= \min_{l \neq l'} \left\{ \frac{1}{n_{l'}} \sum_{i' \in c_{l'}}^{n_{l'}} d(x_i, x_{i'}) \right\} \end{aligned}$$

Sendo assim, temos que a_i indica a média das distância de x_i para as demais observações de C_l, enquanto b_i indica a média das distância de x_i para as observações do *cluster* mais próximo. Fica, então, definido a silhueta s_i da observação x_i, com $i = 1, \ldots, N$, da forma

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}.$$

O coeficiente de Silhueta **S** global será, então:

$$\mathbf{S} = \frac{1}{K} \sum_{l=1}^{K} \frac{1}{n_l} \sum_{i=1}^{n_l} s_i.$$

Dessa forma, para valores de **S maiores** temos agrupamentos mais densos e separados, o que seria o cenário adequado de um agrupamento bem feito.

4. **Calinski-Harabasz (CH)**

   O índice de Calinski e Harabasz Caliński e Harabasz (1974) considera a variância intragrupo, chamada de $WGSS_l$, de cada *cluster* C_l gerado considerando a distância quadrática de cada observação ao seu valor de referência m_l, sendo um centróide ou medóide, de forma que é calculado como

   $$WGSS_l = \sum_{i=1}^{n_l} d^2(x_i, m_l),$$

   assim, a variância intragrupo total $WGSS$ será

   $$WGSS = \sum_{l=1}^{K} WGSS_l.$$

   Para o cálculo da métrica, é utilizada também uma medida de dispersão $BGSS$ entre os grupos, obtida através da soma ponderada das distâncias quadráticas do valor de referência de cada *cluster* e um valor central global m, o que se dá pela fórmula abaixo:

   $$BGSS = \sum_{l=1}^{K} n_l d^2(m_l, m).$$

   Assim, o índice **CH** de Calinski e Harabasz será obtido por:

   $$\mathbf{CH} = \frac{N - K}{K - 1} \times \frac{BGSS}{WGSS},$$

   que apresenta valores **maiores** quando temos grupos com menor variância e bem separados.

3 Aplicação

Nesse capítulo será realizada a aplicação da teoria desenvolvida ao longo do Capítulo 2. Os dados utilizados para o estudo são apresentados na Seção 3.1, cuja análise exploratória é feita na Seção 3.2. Já na Seção 3.3 são apresentados os resultados dos diferentes métodos de agrupamento obtidos. A seleção do agrupamento que melhor se adequa aos dados utilizados de acordo com os índices estudados está na Seção 3.4. Por fim, na Seção 3.5 é feito uma análise dos grupos obtidos pelo agrupamento selecionado.

3.1 Sobre os Dados

As técnicas de agrupamento estudadas serão utilizadas em um conjunto de dados de indicadores obstétricos a nível municipal do ano de 2019 obtido no SINASC, obtidos através da Plataforma de Ciência de Dados aplicada à Saúde (PCDaS) da Fundação Oswaldo Cruz (Fiocruz), disponível em https://pcdas.icict.fiocruz.br/.

Para o entendimento dos indicadores obstétricos utilizados é importante que se tenha conhecimento de conceitos utilizados por eles, os quais são descritos a seguir:

1. Prematuridade: entende-se por parto prematuro aquele que ocorre em idade gestacional do parto inferior a 37 semanas.
2. Gestação Múltipla: gestação que gera dois ou mais fetos simultaneamente.
3. Parto Cesárea: tipo de parto em que a extração do feto é feita por meio de intervenção cirúrgica.
4. Consultas de Pré-Natal: acompanhamento médico realizado pela gestante durante a gravidez. O ideal é ter pelo menos 7 consultas médicas durante o período de pré-natal.
5. Apgar: escala proposta em 1953 pela médica Virgínia Apgar que atribui pontuação a 5 sinais do recém-nascido, sendo eles a frequência cardíaca, respiração, tônus muscular, irritabilidade reflexa e cor da pele. A escala varia de 0 a 10, em que um escore abaixo de 7 merece atenção das(os) obstetras. Essa avaliação é feita no primeiro minuto após o nascimento, o Apgar do primeiro minuto, e após 5 minutos, o Apgar do quinto minuto.
6. Anomalia congênita: alterações estruturais ou funcionais que podem ser causadas por diversos fatores e se têm origem na vida intrauterina.

A seguir é apresentado como foi feita a obtenção dos indicadores obstétricos utilizados, em que as frequências obtidas para o cálculo de cada um deles são referentes ao município de residência dos nascidos vivos no ano de 2019.

Os indicadores obstétricos são obtidos por percentual válido, ou seja, eles são calculados de forma que dados faltantes da variável em questão são desconsiderados.

1. Percentual válido de prematuridade:

$$\frac{\text{número de prematuros}}{\text{número de nascidos vivos com informação de prematuridade}} \times 100$$

2. Percentual válido de gestações múltiplas:

$$\frac{\text{número de gestações múltiplas}}{\text{número de nascidos vivos com informação de gestação múltipla}} \times 100$$

3. Percentual válido de partos cesáreas:

$$\frac{\text{número de cesáreas}}{\text{número de nascidos vivos com informação de tipo de parto}} \times 100$$

4. Percentual válido de nascidos com nenhuma consulta de pré-natal:

$$\frac{\text{número de nascidos com nenhuma consulta de pré-natal}}{\text{número de nascidos vivos com informação de consultas de pré-natal}} \times 100$$

5. Percentual válido de nascidos com 7 ou mais consultas de pré-natal

$$\frac{\text{número de nascidos com 7 ou mais consultas de pré-natal}}{\text{número de nascidos vivos com informação de consultas de pré-natal}} \times 100$$

6. Percentual válido de Apgar de 1º minuto menor que 7

$$\frac{\text{número de nascidos com Apgar de 1º minuto menor que 7}}{\text{número de nascidos vivos com informação de Apgar de 1º minuto}} \times 100$$

7. Percentual válido de Apgar de 5º minuto menor que 7

$$\frac{\text{número de nascidos com Apgar de 5º minuto menor que 7}}{\text{número de nascidos vivos com informação de Apgar de 5º minuto}} \times 100$$

8. Percentual válido de nascidos com anomalia congênita

$$\frac{\text{número de nascidos com anomalia congênita}}{\text{número de nascidos vivos com informação de anomalia congênita}} \times 100$$

9. Percentual válido de nascidos com peso menor que 2.500 gramas

$$\frac{\text{número de nascidos com peso menor que 2.500 gramas}}{\text{número de nascidos vivos com informação de peso no nascimento}} \times 100$$

10. Percentual válido de nascidos do sexo feminino

$$\frac{\text{número de nascidos do sexo feminino}}{\text{número de nascidos vivos com informação de sexo}} \times 100$$

Como a unidade de análise é o município brasileiro, para o cálculo do indicador obstétrico em questão é necessário ter a identificação do município. Ao analisar os dados do SINASC agrupados por município, há 23 observações com "MUNICIPIO IGNORADO" como nome do município. Essas informações podem ser consultadas no Apêndice A, porém foram ignoradas durante o desenvolvimento dessa análise. Seguimos, então com as observações dos 5.570 municípios brasileiros.

Na Tabela 1 estão relacionadas variáveis contidas no banco de dados analisado, assim como o rótulo usado nas análises subsequentes e as respectivas descrições.

variável	descrição
uf	unidade federativa (estados + distrito federal)
municipio	município
codigo	código do município do IBGE
nascidos_vivos	número de nascidos vivos
porc_premat	percentual válido de prematuridade
porc_gesta_multipla	percentual válido de gestações múltiplas
porc_cesarea	percentual válido de partos cesáreas
porc_0_consulta	percentual válido de nascidos com nenhuma consulta de pré-natal
porc_7mais_consulta	percentual válido de nascidos com 7 ou mais consultas de pré-natal
porc_apgar1_menor_7	percentual válido de Apgar de 1º minuto menor que 7
porc_apgar5_menor_7	percentual válido de Apgar de 5º minuto menor que 7
porc_anomalia	percentual válido de nascidos com anomalia congênita
porc_peso_menor_2500	percentual válido de nascidos com peso menor que 2.500 gramas
porc_fem	percentual válido de nascidos do sexo feminino

Tabela 1 – Dicionário das variáveis do conjunto de dados.

Após a análise de agrupamento, os grupos criados serão analisados com relação a indicadores socioeconômicos obtidos pelo Censo Demográfico de 2010, disponíveis em https://basedosdados.org/dataset/mundo-onu-adh. Como esses dados foram coletados 9 anos

antes das taxas obstétricas do estudo, existe uma diferença de 5 municípios que foram criados após o Censo 2010, apresentados na Tabela 2.

Unidade Federativa	Município	Código do IBGE
Pará	Mojuí dos Campos	150475
Santa Catarina	Balneário Rincão	422000
Santa Catarina	Pescaria Brava	421265
Rio Grande do Sul	Pinto Bandeira	431454
Mato Grosso do Sul	Paraíso Das Águas	500627

Tabela 2 – Municípios criados após o Censo 2010

Os indicadores socioeconômicos considerados estão listados abaixo:

1. Fecundidade total: Número médio de filhos nascidos vivos, tidos por uma mulher ao final do seu período reprodutivo, na população residente em determinado espaço geográfico, no ano considerado.

2. Índice de Gini: índice que varia de 0 a 1 desenvolvido como medida de desigualdade, em que 0 corresponde à completa igualdade (no caso do rendimento, por exemplo, toda a população recebe o mesmo salário) e 1 corresponde à completa desigualdade (onde uma pessoa recebe todo o rendimento e as demais nada recebem).

3. Renda per capita média: é calculada pela razão entre o PIB (Produto Interno Bruto) e o número de habitantes do país. O PIB é a soma do valor de tudo que foi produzido no país por suas empresas durante o período de um ano.

4. Índice de desenvolvimento humano municipal (IDHM): é uma medida que varia entre 0 e 1 e leva em consideração três dimensões do desenvolvimento humano: longevidade (mede a expectativa de vida da população), educação (mede o acesso ao conhecimento) e a renda (mede o padrão de vida). Quanto maior for o IDHM, maior é o desenvolvimento do município.

3.2 Análise Exploratória dos Dados

As medidas resumo do número de nascidos vivos e dos indicadores obstétricos dos municípios brasileiros podem ser vistas na Tabela 3. Observamos que a variável de nascidos vivos tem uma amplitude muito alta, em que varia de município com 2 nascidos vivos no ano de 2019 até 158.587, que correspondem, respectivamente, às cidades de Cedro do Abaeté, localizada no estado de Minas Gerais, e São Paulo, capital do estado de São Paulo.

Ainda sobre o número de nascidos vivos por município, temos que 76 municípios tiveram 5.000 ou mais nascidos vivos em 2019. Desses, 17 estão localizados no estado de São Paulo.

Enquanto no outro extremo, com menos de 50 nascidos vivos no ano, temos 1001 municípios, em que se destacam Rio Grande do Sul, com 227 municípios nessa categoria e em seguida Minas Gerais com 207. A distribuição do número de nascidos vivos por município brasileiro pode ser observada no mapa da Figura 7, utilizando uma escala logarítmica para facilitar a visualização.

Variável	média	dp	med	min	max
nascidos_vivos	511,50	2844,57	144,00	2,00	158.587,00
porc_premat	11,04	4,34	10,83	0,00	62,95
porc_gesta_multipla	2,04	2,20	1,75	0,00	25,00
porc_cesarea	60,08	16,96	60,00	5,62	100,00
porc_0_consulta	1,18	2,91	0,58	0,00	91,61
porc_7mais_consulta	76,01	13,70	78,84	3,73	100,00
porc_apgar1_menor_7	5,64	3,84	5,16	0,00	66,02
porc_apgar5_menor_7	1,07	1,34	0,84	0,00	30,51
porc_anomalia	0,80	1,14	0,53	0,00	18,75
porc_peso_menor_2500	8,19	3,47	8,01	0,00	37,50
porc_fem	48,79	5,58	48,84	0,00	83,33

Tabela 3 – Medidas descritivas dos indicadores obstétricos e número de nascidos vivos por município brasileiro, sendo elas a média (média), desvio padrão (dp), mediana (med), mínimo (min) e máximo (max).

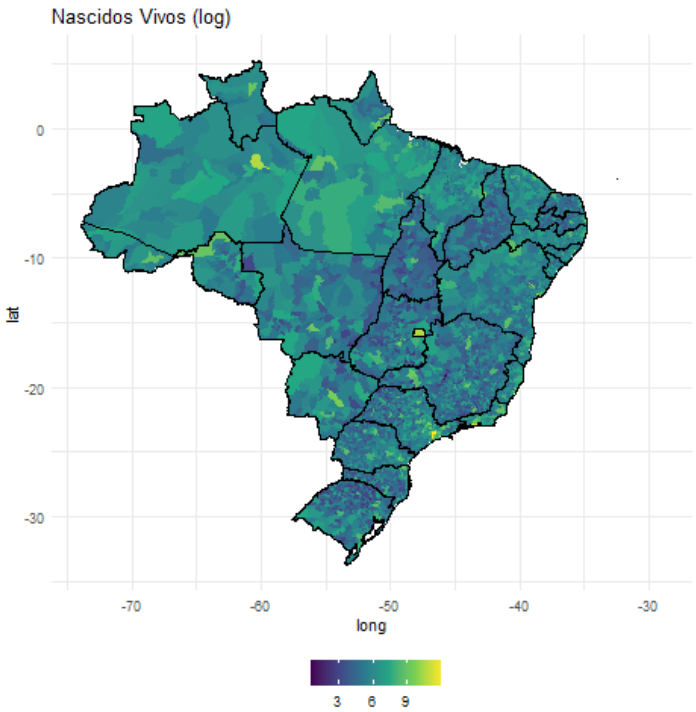
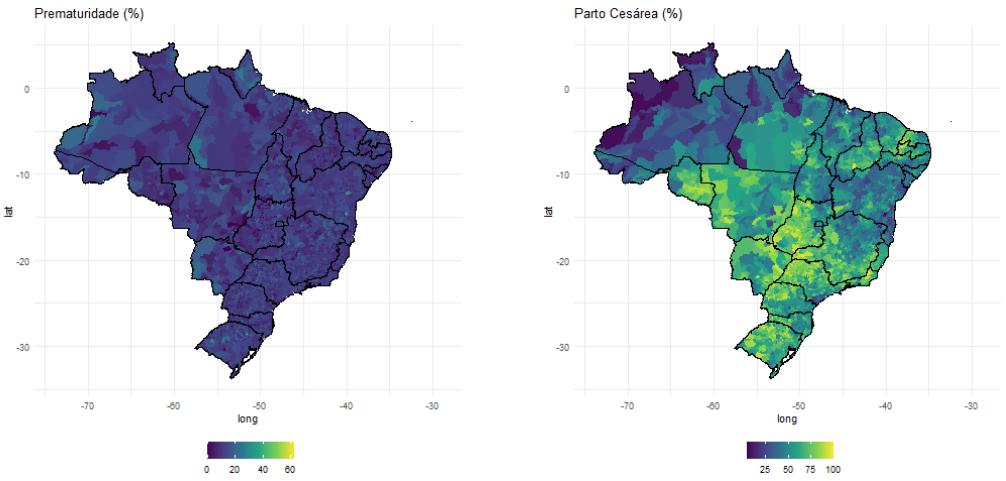


Figura 7 – Mapa dos municípios para a variável número de nascidos vivos (aplicação da transformação logarítmica).

Ao analisar os indicadores obstétricos, que podem ser visualizados também nos gráficos no Apêndice B, vemos que a maior porcentagem de prematuridade apresentou valor de 62,95%

no município de Palmeirândia, no Maranhão, tendo apresentado um total de 310 nascidos vivos, o que é um dado preocupante por ser um alto índice de prematuridade. Na Figura 8(a) essa cidade se destaca em amarelo no mapa que apresenta pouca variabilidade nesse indicador. Outra cidade que apresenta alto percentual de prematuridade é Iguaí, na Bahia, com taxa de 37,5% e total de 430 nascidos vivos no ano de 2019.

Outro valor que chama atenção é o percentual de 100% de partos cesáreas, que ocorre em 17 municípios, sendo que nesses municípios o número de nascidos vivos varia de 8 a 50, tendo, também, o município de Itaocara no Rio de Janeiro que se destacou por apresentar 97,84% de taxa de cesarianas, tendo tido 279 nascidos vivos. Observando o mapa na Figura 8(b), observamos que existe uma distribuição geográfica desse indicador, em que municípios da região Norte apresentam, com exceção de Rondônia, menores taxas de cesarianas. Dentre os estados do Nordeste, destaca-se o estado da Bahia, que conta com o município de Iguaí, que tem a menor taxa de partos cesáreas do país. No outro extremo está a região central do país, com destaque para os municípios do estado de Goiás, com altos índices de cesáreas nesse ano.



(a) Porcentagem de prematuridade.

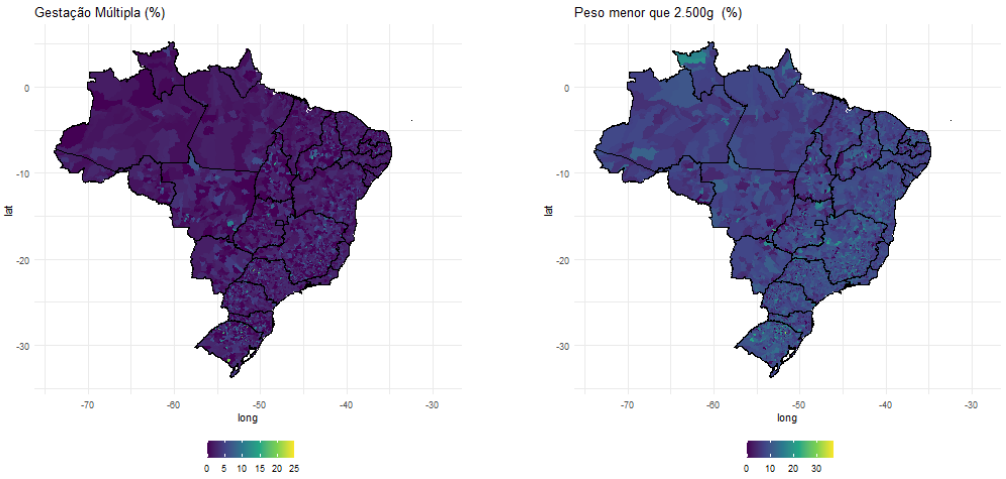
(b) Porcentagem de partos cesáreas.

Figura 8 – Mapas dos municípios para os indicadores de prematuridade e parto cesárea.

A variável de porcentagem de gestações múltiplas apresenta pouca variação entre os municípios, como pode ser visualizado na Figura 9(a), em que os maiores valores observados são de cidades com baixo número de nascidos vivos: 25% em Anhanguera, em Goiás, com 8 nascidos vivos no ano de 2019, e 20% nas cidades de Pedras Altas, Salvador das Missões e Tupanci do Sul, todas no estado do Rio Grande do Sul, com no máximo 20 nascidos vivos no ano.

Em relação ao peso dos nascidos vivos, na Figura 9(b) vemos que os municípios de Alto Alegre e Amajari, no norte do estado de Roraima, se destacam dos demais municípios da região, com 18,57% e 16,59% de nascidos vivos com peso abaixo de 2.500g. O maior percentual observado para essa variável foi, assim como na porcentagem de gestação múltipla, no município goiano de Anhanguera. Os demais municípios com taxas superiores a 25% são Centenário, Entre

Rios do Sul, Floriano Peixoto, Guabiju e Ivorá, todos do estado do Rio Grande do Sul.



(a) Porcentagem de gestações múltiplas.

(b) Porcentagem de nascidos vivos com peso menor que 2.500g.

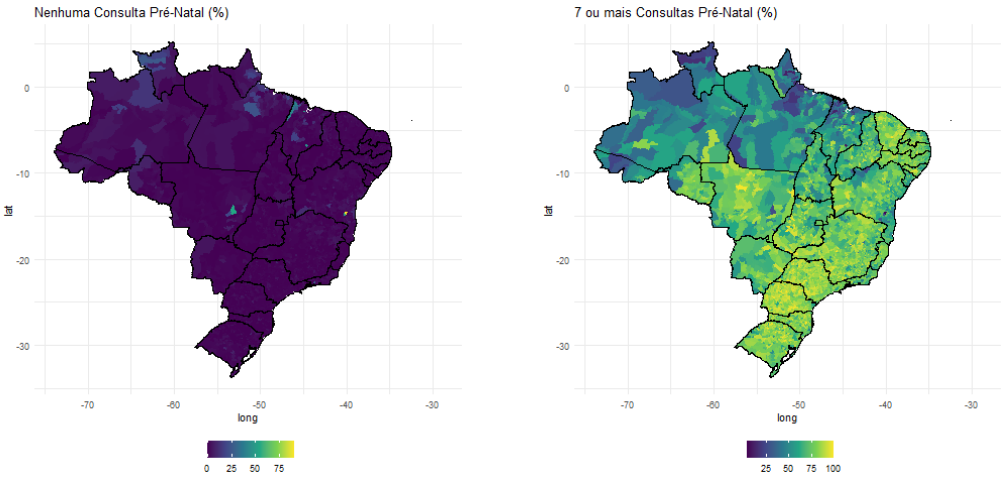
Figura 9 – Mapas dos municípios para os indicadores de gestação múltipla e peso.

A porcentagem de nascidos vivos que não tiveram nenhuma consulta de pré-natal no município de Iguaí, no estado da Bahia, foi de 91,61%, o que é uma taxa extremamente preocupante, dada a importância do acompanhamento pré-natal na garantia de saúde da gestante e do feto. Observando o mapa da Figura 10(a) algumas outras cidades também se destacam, como é o caso de Campinápolis, no Mato Grosso, com porcentagem de 54,59%, e de municípios na região Norte, em especial no estado do Maranhão.

Dos 5570 municípios brasileiros, 2141 não contaram com nascidos vivos sem consulta de pré-natal, desses, 2 apresentaram mais de 1.000 nascidos vivos de mães residentes: Cachoeiro de Itapemirim, no Espírito Santo, com 2.579, e Poços de Caldas, no estado de Minas Gerais, com 1.904 registros de nascidos vivos.

Já a distribuição da porcentagem de nascidos vivos que tiveram ao menos 7 consultas de pré-natal, como podemos ver na Figura 10(b), apresenta valores mais baixos nas regiões citadas para os municípios com altas taxas de nascidos vivos sem consulta de pré-natal, em que temos Iguaí com o menor percentual de 7 ou mais consultas pré-natal do país, seguido por municípios da região norte. Vale, entretanto, ressaltar que alguns municípios apresentaram baixas porcentagens em ambos os indicadores, como é o caso de Breves, no Pará, com 16,31% sem consulta pré-natal e 17,53% com 7 ou mais. De todas as cidades, 38 tiveram 100% de seus nascidos vivos com ao menos 7 consultas pré-natal, a com maior número de nascimentos, entretanto, registrou apenas 40 nascidos vivos.

As porcentagens de nascidos vivos com Apgar menor que 7 observado no primeiro e no quinto minuto apresentam comportamento similar, de baixa variabilidade, com redução do percentual no 5º minuto quando comparado ao observado no 1º; como é o caso dos municípios

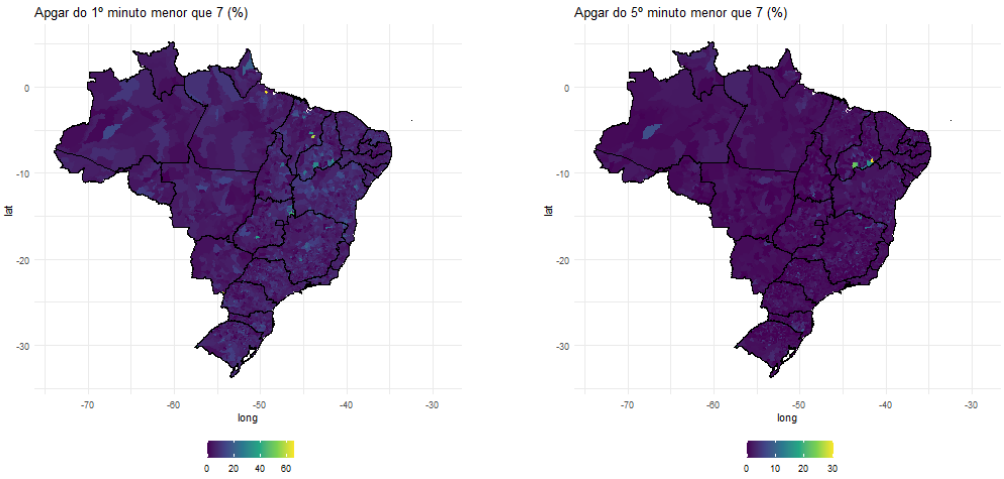


(a) Porcentagem de nascidos vivos sem nenhuma consulta pré-natal.

(b) Porcentagem de nascidos vivos com 7 ou mais consultas pré-natal.

Figura 10 – Mapas dos municípios para os indicadores de número de consultas de pré-natal.

de Santa Cruz do Arari, no Pará, Mundo Novo, no Mato Grosso do Sul, e Buriti Bravo, no Maranhão, que apresentaram as maiores porcentagens de Apgar do primeiro minuto abaixo de 7, 66,02%, 61,49% e 57,74%, respectivamente. O percentual de Apgar do quinto minuto abaixo de 7 desses municípios diminuiu para abaixo de 5%, queda que não foi tão forte nos 3 municípios com maiores taxas de Apgar do quinto minuto abaixo de 7, todos do Piauí.



(a) Porcentagem de Apgar do 1º minuto menor que 7.

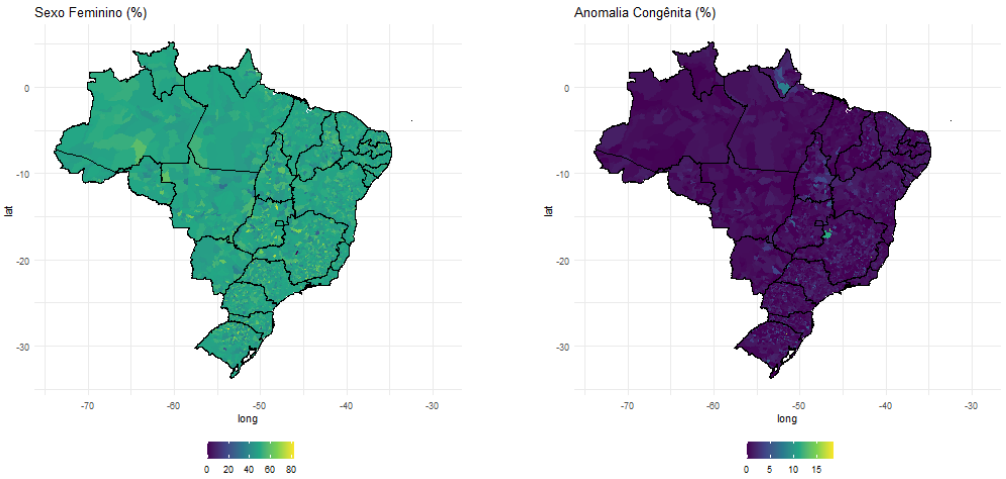
(b) Porcentagem de Apgar do 5º minuto menor que 7.

Figura 11 – Mapas dos municípios para os indicadores de Apgar.

As porcentagens de nascidos vivos com alguma anomalia congênita são baixas para todos os municípios, como vemos na Figura 12(b), em que alguns estados se destacam com índices um pouco maiores, como é o caso de cidades do Amapá, Tocantins e Goiás. Temos, além disso, os municípios que apresentam as maiores porcentagens, sendo em grande parte lugares com baixo número de nascidos vivos, como Ponte Preta e André da Rocha, ambos do Rio Grande do Sul,

com 18,75% e 16,67% de anomalia e 16 e 6 nascidos vivos, respectivamente. Dois municípios com mais de 1.000 nascidos vivos apresentaram taxas de anomalia maior que 8%: Santana, no Amapá, e Paracatu, em Minas Gerais, com 8,55% e 11,43%, respectivamente. Vale notar os dois municípios mineiros que se destacam no mapa, Taparuba e Natalândia, com 15,00% e 14,28% de anomalia, respectivamente, ambos com 37 nascidos vivos.

Por fim, observamos o maior índice de nascimento de bebês do sexo feminino na cidade de Nova Ramada no Rio Grande do Sul, que apresentou 12 nascimentos.No outro extremo temos as cidades mineiras de Cedro do Abaeté, com 2 nascimentos, e Serra da Saudade, com 6 nascimentos, ambas sem nascidos vivos do sexo feminino. Já as menores porcentagens diferentes de 0% são das cidades de Ivorá e Travesseiro, ambas do Rio Grande do Sul, e São Francisco, do estado de São Paulo, com 14,29%, 18,18% e 18,18% de nascidos vivos do sexo feminino, respectivamente. O mapa para a porcentagem de sexo feminino está na Figura 12(a).



(a) Porcentagem de nascidos vivos do sexo feminino.

(b) Porcentagem de nascidos vivos com anomalia congênita.

Figura 12 – Mapas dos municípios para os indicadores de sexo feminino e anomalia congênita.

Na Figura 13 temos as correlações entre os pares de indicadores obstétricos, que foram obtidas pelos pacotes `stats` (Wei e Simko, 2017). Para o cálculo de correlação foi considerado o coeficiente de correlação de postos de Spearman. Esse coeficiente mede a correlação, não necessariamente linear, entre duas variáveis, além de não assumir qualquer pressuposto a respeito da distribuição de ambas as variáveis. Dessa forma, valores de coeficiente de correlação de Spearman, que varia de -1 a 1, próximos a 1 implicam em um tendência de crescimento mútuo entre as variáveis observadas, enquanto valores próximos a -1 indicam que a medida que aumentamos os valores de uma das variáveis, a outra tende a decrescer.

Um primeiro destaque é o fato de que todas correlações são menores que 0,5 em módulo. Podemos observar que a maior correlação observada nos dados é entre a porcentagem de nascidos vivos com peso menor que 2.500 gramas e a porcentagem de prematuridade, com coeficiente de 0,46, uma correlação moderada. Em seguida temos a porcentagem de nascidos com nenhuma

consulta pré-natal e número de nascidos vivos, com coeficiente de correlação de 0,39, que se traduz em maior a porcentagem de nascimentos sem acompanhamento pré-natal à medida que aumentamos o número de nascidos vivos.

Relações positivas moderadas também ocorrem entre a porcentagem de cesáreas e a porcentagem de nascidos com 7 ou mais consultas de pré-natal, e entre a porcentagem de gestações múltiplas e percentual de nascidos vivos com menos de 2.500g. Além da correlação moderada entre as porcentagens de Apgar do primeiro e do quito minuto menores que 7.

Dentre as correlações negativas destaca-se a relação entre o percentual de nascidos com nenhuma consulta pré-natal e o percentual com 7 ou mais, o que é esperado. Seguida pela correlação de -0,28 entre a porcentagem de parto cesárea e a porcentagem de nascidos que não foram acompanhados em nenhuma consulta pré-natal, o que condiz com a correlação positiva moderada entre a percentual com 7 ou mais consultas e a taxa de cesarianas.

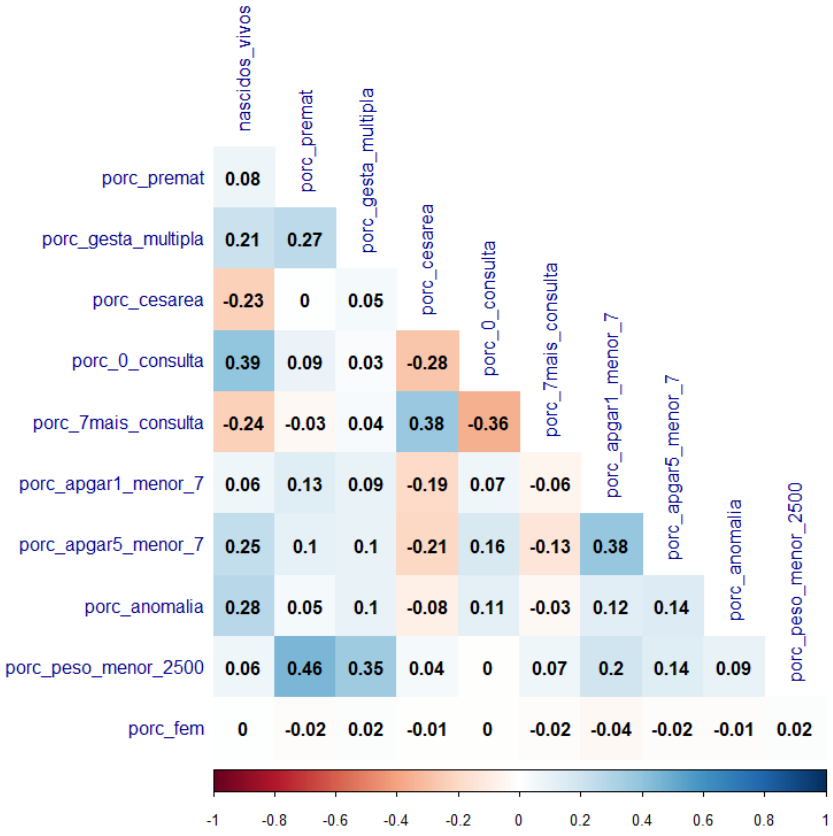


Figura 13 – Correlograma dos indicadores obstétricos e número de nascidos vivos.

3.3 Análise de Agrupamento

Nesta seção será desenvolvida a análise de agrupamento proposta na Seção 2. Os métodos de agrupamento por particionamento considerados são aplicados na Seção 3.3.1, os ajustes dos métodos hierárquicos estão na Seção 3.3.2, as aplicações dos métodos DBSCAN e agrupamento

espectral estão dedicadas nas seções 3.3.3 e 3.3.4, respectivamente. Na Seção 3.4 está a discussão de qual método apresentou melhor resultado, baseada nos métodos de validação descritos em 2.3.

Para as análises de agrupamento, são considerados os indicadores obstétricos: porcentagem de prematuridade, porcentagem de gestações múltiplas, porcentagem de partos cesáreas, porcentagem de nascidos vivos com nenhuma consulta de pré-natal, porcentagem de nascidos vivos com 7 ou mais consultas de pré-natal, porcentagem de nascidos vivos com Apgar de primeiro e quinto minuto menor que 7, porcentagem de anomalias congênias, porcentagem de nascidos vivos com peso ao nascer menor que 2.500 gramas e porcentagem de nascidos vivos do sexo feminino. Ainda para a análise, esses indicadores obstétricos são padronizados, ou seja, subtraídos pela sua média e divididos pelo desvio-padrão.

O desenvolvimento computacional foi realizado por meio do R (R Core Team, 2021) e os pacotes utilizados são mencionados nas descrições das aplicações dos métodos em questão.

3.3.1 Agrupamentos por Particionamento

Como apresentado na Seção 2.2.1, os métodos por particionamento necessitam do número de grupos pré estabelecido. Essa decisão foi feita baseada na análise visual no seu gráfico do cotovelo, através do pacote `factoextra` (Kassambara e Mundt, 2020). O método por particionamento K-médias foi implementado utilizando o pacote `stats` (R Core Team, 2021), considerando o algoritmo de Hartigan e Wong (1979), enquanto os métodos PAM e CLARA utilizados são os disponíveis no pacote `cluster` (Maechler et al., 2021).

Na Figura 14 observamos que para os 3 métodos de particionamento o número de grupos $K = 7$ se mostrou ótimo, em que a partir dele, não diminui mais o valor da variância de forma relevante.

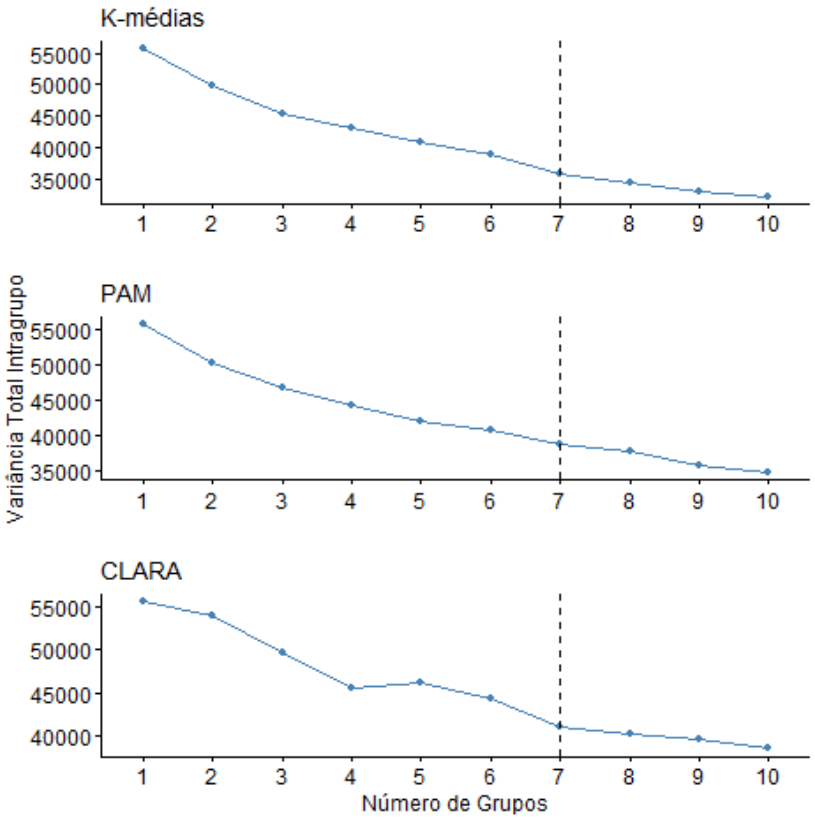


Figura 14 – Gráficos do método do cotovelo para os agrupamentos por particionamento

Os agrupamentos resultantes dos métodos K-médias, PAM e CLARA podem ser observados nas Tabelas 4,5 e 6, respectivamente.

		Centróide (valores padronizados)									
Grupo	n	Prematuridade	Gestação Múltilpla	Parto Cesárea	0 Consulta Pré-Natal	7+ Consultas Pré-Natal	Apgar 1º min menor que 7	Apgar 5º min menor que 7	Anomalia Congênita	Peso Menor que 2.500g	Sexo Feminino
1	573	-0,84	-0,54	0,23	-0,16	0,33	-0,37	-0,37	-0,33	-0,94	-1,26
2	1587	0,33	0,13	0,60	-0,15	0,40	-0,21	-0,23	-0,06	0,30	-0,10
3	1387	0,00	-0,11	-0,74	-0,04	-0,16	0,71	0,58	-0,02	0,08	0,00
4	936	-0,72	-0,45	0,30	-0,17	0,26	-0,50	-0,40	-0,26	-0,66	0,85
5	404	1,38	2,09	0,38	-0,15	0,28	0,30	0,30	-0,08	1,63	0,20
6	206	-0,01	-0,17	0,21	-0,13	0,20	-0,02	0,04	3,35	0,06	0,02
7	477	0,15	-0,29	-1,16	1,31	-2,10	-0,18	0,04	-0,23	-0,22	0,00

Tabela 4 – Tabela resumo dos grupos obtidos pelo agrupamento K-médias.

Grupo	n	Medóide (município - UF)
1	1203	Rio Claro - SP
2	712	Santa Maria de Jetibá - ES
3	335	Mara Rosa - GO
4	1324	Palhoça - SC
5	847	Rincão - SP
6	744	São Cristóvão - SE
7	405	Nova Prata do Iguaçu - PR

Tabela 5 – Tabela resumo dos grupos obtidos pelo agrupamento PAM.

Grupo	n	Medóide (município - UF)
1	2150	Araguari - MG
2	2403	Brasília - DF
3	871	Santo Antônio do Descoberto - GO
4	81	Mata - RS
5	15	Barra do Quaraí - RS
6	35	Iguatu - PR
7	15	Coronel Pilar - RS

Tabela 6 – Tabela resumo dos grupos obtidos pelo agrupamento CLARA.

3.3.2 Agrupamentos Hierárquicos

Na Seção 2.2.2 foram apresentados variados métodos hierárquicos. Nesse estudo consideramos os 5 diferentes métodos aglomerativos demonstrados, utilizando o pacote `stats`, enquanto o método divisivo DIANA foi implementado pela função disponível no pacote `cluster`.

Como discutido, a seleção do número de grupos para esses agrupamentos é comumente feita com base no seu dendrograma, porém, como pode ser visto no Apêndice C, nenhum método apresenta um dendrograma com divisão clara de melhor número de grupos particionar os dados, com exceção do método de Ward. Por isso, seguiremos considerando valores de $K \in \{3, \ldots, 8\}$, em que o resultado está disposto na tabela 7.

K	Grupo	Metodo					
		Média das Distâncias	Centróide	Vizinho mais Distante	Vizinho mais Próximo	Ward	DIANA
3	1	5566	5566	5547	5568	4418	5539
	2	3	3	20	1	1140	18
	3	1	1	3	1	12	13
4	1	5565	5565	5547	5567	2089	5539
	2	3	3	19	1	2329	17
	3	1	1	3	1	1140	13
	4	1	1	1	1	12	1
5	1	5564	5564	5537	5566	2089	5515
	2	3	1	19	1	2329	17
	3	1	3	3	1	513	13
	4	1	1	1	1	627	24
	5	1	1	10	1	12	1
6	1	5553	5561	5529	5565	1915	5515
	2	11	1	19	1	2329	17
	3	3	3	8	1	513	10
	4	1	1	3	1	627	24
	5	1	3	1	1	174	3
	6	1	1	10	1	12	1
7	1	5552	5557	5524	5562	1915	5399
	2	11	4	19	3	2162	116
	3	1	1	8	1	513	17
	4	3	3	5	1	167	10
	5	1	1	3	1	627	24
	6	1	3	1	1	174	3
	7	1	1	10	1	12	1
8	1	5541	5556	5524	5561	1405	5399
	2	11	4	19	3	2162	115
	3	1	1	7	1	510	17
	4	3	3	5	1	513	10
	5	1	1	3	1	167	24
	6	11	3	1	1	627	3
	7	1	1	10	1	174	1
	8	1	1	1	1	12	1

Tabela 7 – Distribuição do número de observações por *cluster* dos agrupamentos hierárquicos.

O método BIRCH, descrito na Seção 2.2.2.1, não possui implementação pronta em uma função em pacote na linguagem R, porém, foi encontrada um implementação em documentação disponível no GitHub (https://github.com/rohitkata/BIRCH-Clustering-R-package).

Para o BIRCH, foram estabelecidos os parâmetros de fator de ramificação $B = 4$ e limiar $T = 3$, sendo o valor de T escolhido baseado na distância de cada ponto para seus 3 vizinhos mais próximos. Com esses parâmetros obtivemos uma CF-*tree* com 71 nós.

Subgrupo	n	Subgrupo	n	Subgrupo	n	Subgrupo	n	Subgrupo	n	Subgrupo	n
1	1523	13	1	25	16	37	2	49	3	61	1
2	230	14	1	26	1	38	1	50	8	62	1
3	2603	15	3	27	3	39	10	51	2	63	1
4	568	16	1	28	3	40	1	52	5	64	1
5	4	17	1	29	1	41	1	53	2	65	1
6	6	18	2	30	6	42	2	54	3	66	5
7	32	19	3	31	4	43	2	55	1	67	1
8	9	20	6	32	2	44	2	56	1	68	1
9	1	21	414	33	4	45	1	57	1	69	2
10	19	22	1	34	5	46	1	58	1	70	1
11	2	23	2	35	16	47	1	59	1	71	1
12	3	24	1	36	2	48	2	60	1	-	-

Tabela 8 – Número de observações por subgrupo resultante do agrupamento local do método BIRCH.

Na Tabela 8 vemos um comportamento parecido com os métodos hierárquicos abordados anteriormente, em que temos vários grupos, subgrupos no caso da CF-*tree*, com poucas observações, o que mostra um problema da abordagem hierárquica para os dados de indicadores obstétricos trabalhados, que apresentam pouca variabilidade.

A partir do resultado obtido pelo agrupamento local realizado, utilizamos o método K-médias para o agrupamento global com os subgrupos estabelecidos. Para esse novo agrupamento é necessário a seleção do número K de *clusters*, que foi selecionado pelo método do cotovelo.

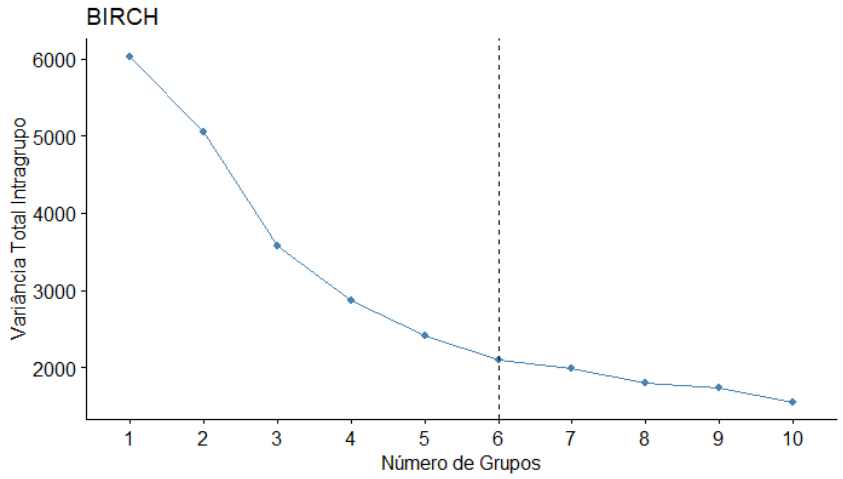


Figura 15 – Gráficos do método do cotovelo para o agrupamento BIRCH.

Na Figura 15 temos a seleção de $K = 6$ como o melhor número de partições para os dados dos subgrupos fornecidos pela CF-*tree*.

Dessa forma temos o resultado final para o agrupamento pelo método BIRCH na Tabela 9. Nela vemos que os três municípios pertencentes ao grupo 1 as três cidades piauienses de

Grupo	1	2	3	4	5	6
n	3	34	65	14	14	5440

Tabela 9 – Número de observações por *cluster* do agrupamento global do método BIRCH.

Caracol, Guaribas e Lagoa do Barro do Piauí, que, como pode ser observado no Apêndice B, apresentam as maiores taxas de Apgar do 5º minuto menor que 7.

Os algoritmos desenvolvidos na Seção 2.2.2.2 não foram implementados. O Chameleon não foi utilizado pela ausência de implementação na linguagem R, que foi utilizada em todo o trabalho. Já o método ROCK não foi implementado por estarmos lidando com um conjunto de dados que possui apenas variáveis numéricas, o que seria a justificativa de seu uso.

3.3.3 Agrupamento por DBSCAN

O método de agrupamento por densidade DBSCAN apresentado na Seção 2.2.3 foi implementado utilizando as funções do pacote `dbscan` (Hahsler e Piekenbrock, 2022).

Como parâmetros para o método precisamos informar um valor de $MinPts$, em que definimos um $MinPts = 4$, assim, construímos o gráfico de distâncias para os $MinPts - 1$ pontos mais próximos para todas as observações para definir o parâmetro ϵ.

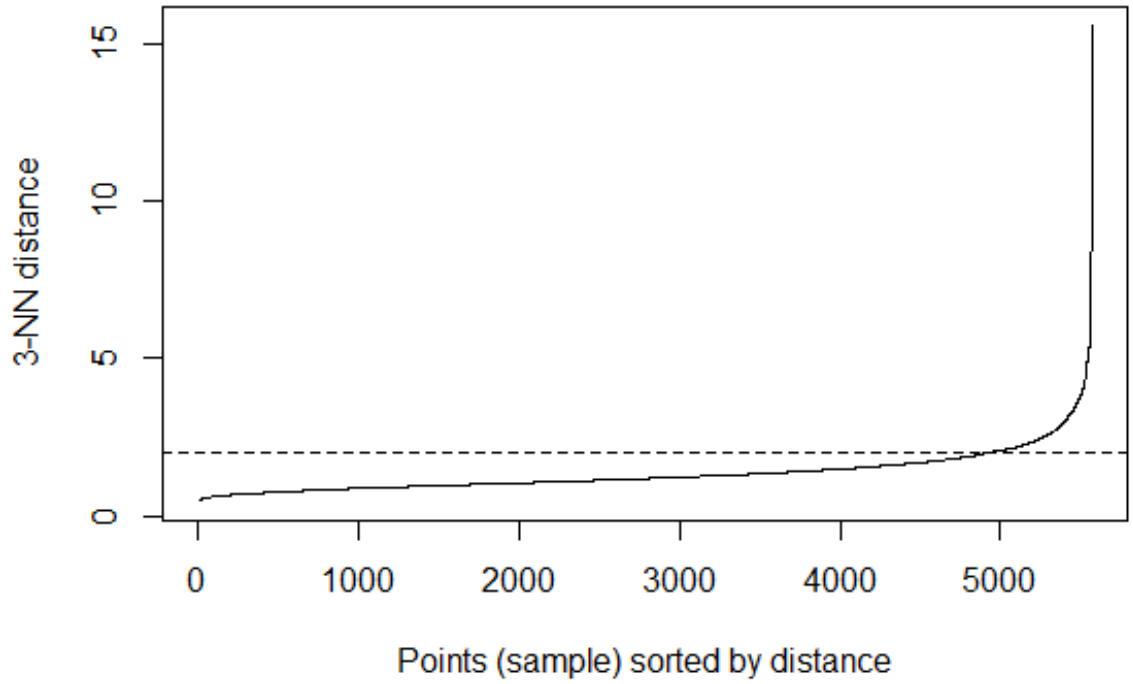


Figura 16 – Distância para os 3 vizinhos mais próximos do método DBSCAN.

Observando o gráfico da Figura 16 temos que $\epsilon = 2$ é um valor adequando. Realizando, então, o agrupamento pelo método DBSCAN com os parâmetros definidos, foi obtido o resultado da Tabela 10.

Grupo	1	2	3	4	5	*outlier*
n	5102	4	4	3	5	452

Tabela 10 – Número de observações por *cluster* do agrupamento DBSCAN.

Observamos que 452 das observações dos dados foram identificadas como *outliers*, e, com o dado restante, a grande maioria ficou em um grupo apenas. Notamos um grupo com um número de objetos menor que o $MinPts$ definido, o que se explica pela existência de uma observação que é Ponto de Borda a mais de um *cluster*, o que a faz ser alocada no grupo que a qual foi designada primeiro.

3.3.3.1 Análise *Outliers*

A listagem de todos os municípios considerados *outliers* pelo DBSCAN pode ser consultada no Apêndice D. É importante notar que nenhuma capital estadual ou a capital federal

foi considerada *outlier* pelo método. O estado do Espírito Santo foi a única unidade federativa, fora o Distrito Federal, que não apresentou nenhum município classificado como *outlier*, enquanto o estado do Amapá foi o que apresentou maior número, proporcionalmente, de municípios como *outliers*, sendo 7 de suas 16 cidades classificadas como tal.

Na Tabela 11 podemos observar as medidas descritivas das variáveis das observações tidas como *outliers*. Na Figura 17 vemos que os municípios identificados como *outliers* tiveram, em grande maioria, um número menor de número de nascidos vivos.

Variável	média	dp	med	min	max
nascidos_vivos	88,48	191,23	40,00	2,00	2230,00
porc_premat	14,36	8,02	13,82	0,00	62,95
porc_gesta_multipla	3,78	4,61	2,13	0,00	25,00
porc_cesarea	61,25	19,72	61,10	5,62	100,00
porc_0_consulta	2,71	7,88	0,00	0,00	91,61
porc_7mais_consulta	73,96	19,13	79,33	3,73	100,00
porc_apgar1_menor_7	9,11	8,36	7,23	0,00	66,02
porc_apgar5_menor_7	2,42	3,29	1,47	0,00	30,51
porc_anomalia	1,71	2,72	0,00	0,00	18,75
porc_peso_menor_2500	11,18	6,46	10,79	0,00	37,50
porc_fem	49,03	10,80	49,02	0,00	83,33

Tabela 11 – Medidas descritivas dos indicadores obstétricos e nascidos vivos por município brasileiro identificado como *outlier* pelo DBSCAN, sendo elas a média (média), desvio padrão (dp), mediana (med), mínimo (min) e máximo (max).

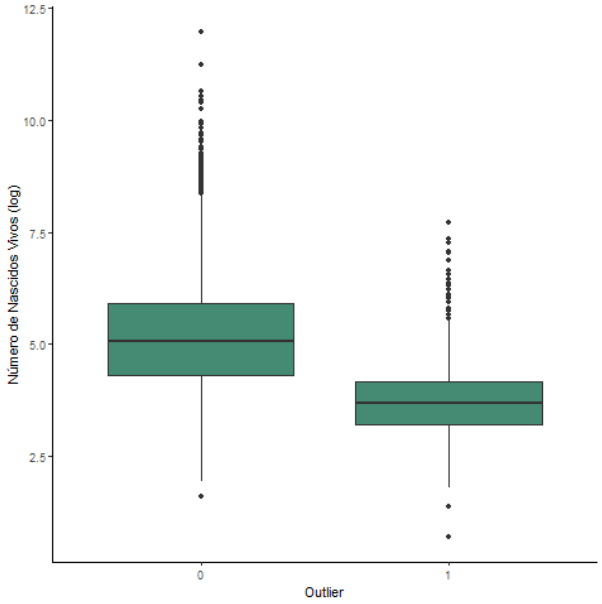
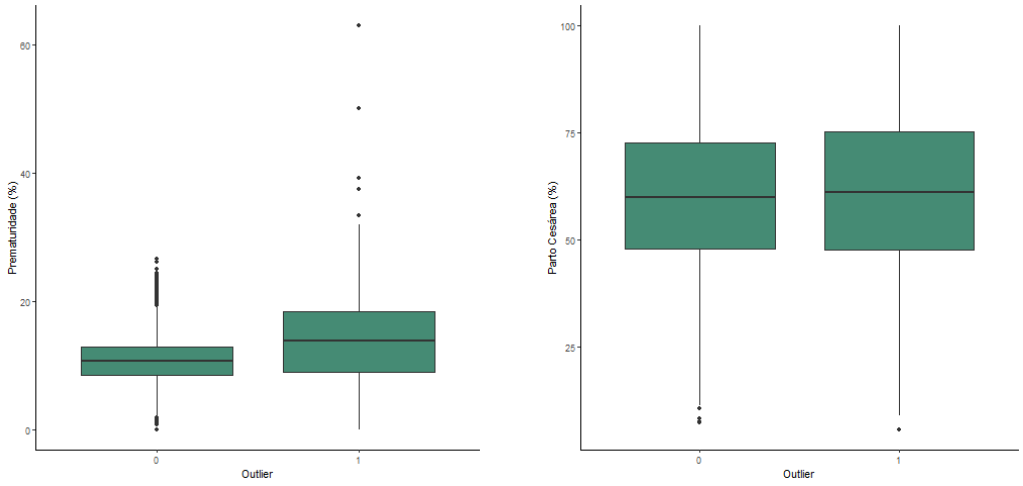


Figura 17 – Boxplot do número de nascidos vivos por município do grupo *outlier* pelo DBSCAN (grupo 1) e não *outlier* (grupo 0).

Comparando as porcentagens de prematuridade, na Figura 18(a), temos que os *outliers* têm uma distribuição com maiores porcentagens de prematuridade. Já a distribuição da porcentagem

de partos cesáreas não apresenta diferença entre os dois grupos, como apresentado na Figura 18(b), com boxplots praticamente idênticos.

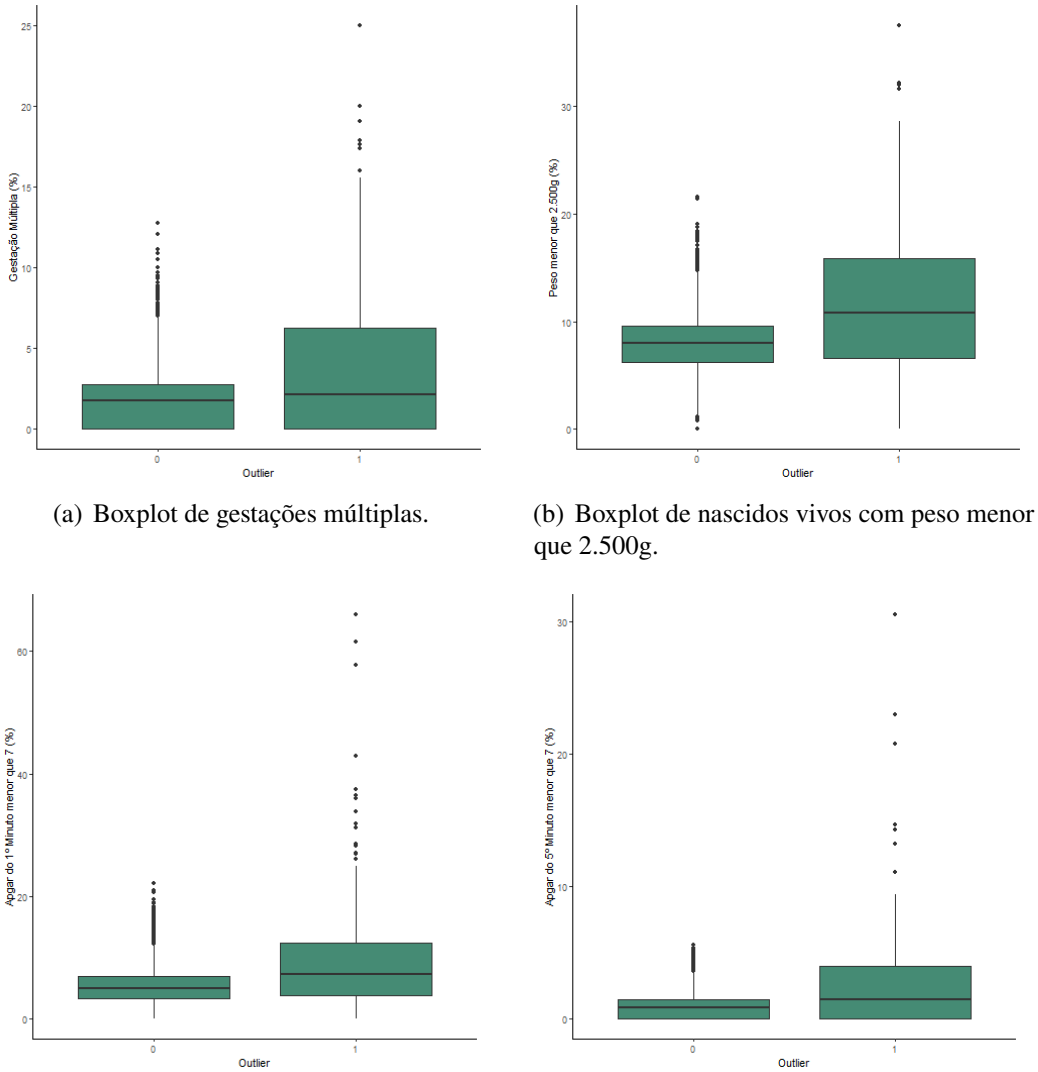


(a) Boxplot de prematuridade.

(b) Boxplot de partos cesáreas.

Figura 18 – Boxplot para as porcentagens de prematuridade e de parto cesárea do grupo *outlier* pelo DBSCAN (grupo 1) e não *outlier* (grupo 0).

Na Figura 19 vemos que em todas as variáveis os *outliers* contém o municípios com maiores taxas, o que seriam os *ouliers* dessas variáveis, além de apresentarem distribuições com valores mias altos.



(a) Boxplot de gestações múltiplas.

(b) Boxplot de nascidos vivos com peso menor que 2.500g.

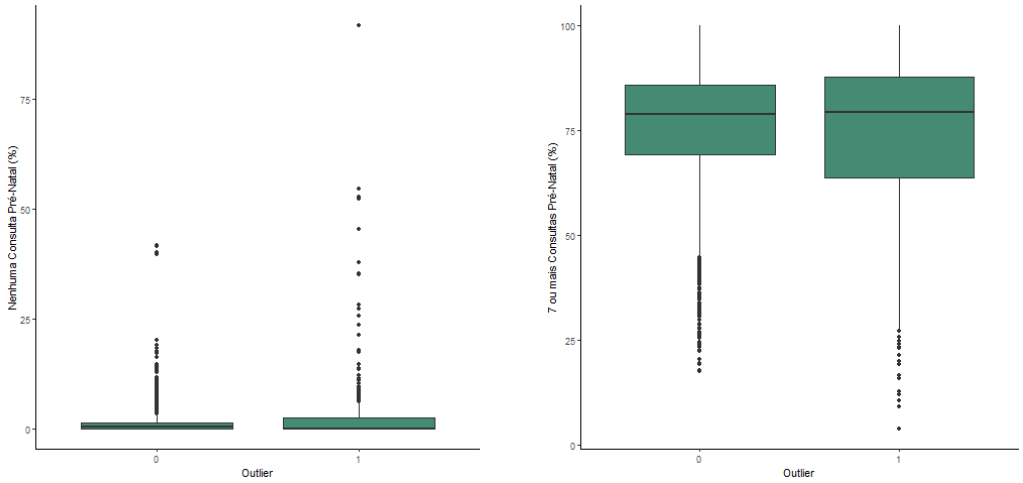
(c) Boxplot de Apgar do 1º minuto menor que 7.

(d) Boxplot de Apgar do 5º minuto menor que 7.

Figura 19 – Boxplot para as porcentagens de gestação múltipla, peso e Apgar do grupo *outlier* pelo DBSCAN (grupo 1) e não *outlier* (grupo 0).

Para as variáveis de número de consultas de pré-natal temos que valores mais extremos foram classificados como *outliers*, porém, ainda temos presentes municípios com porcentagens altas de nenhuma consulta pré-natal e porcentagens baixas de 7 ou mais consultas, o que pode ser observado na Figura 20.

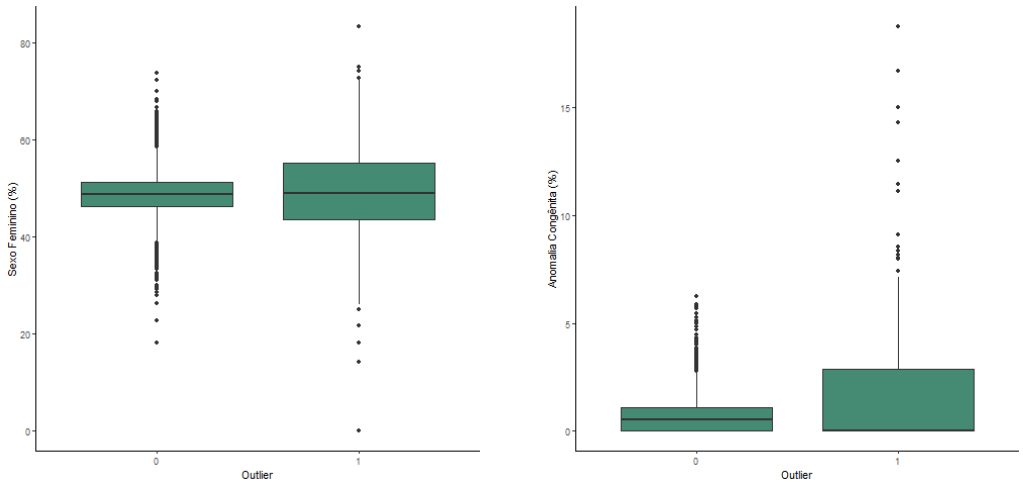
As cidades com as porcentagens de anomalia congênita mais extremas foram consideradas *outliers*, como podemos ver na Figura 21(b), de forma que sua distribuição foi maior para esse grupo. Já as porcentagens de nascidos vivos do sexo feminino, com exceção dos municípios nos extremos da variável, não se mostraram diferentes entre os grupos, os municípios rotulados como *outliers*, porém, apresentam maior variância nessa variável o que é visível na Figura 21(a).



(a) Boxplot de nascidos vivos sem nenhuma consulta pré-natal.

(b) Boxplot de nascidos vivos com 7 ou mais consultas pré-natal.

Figura 20 – Boxplot para as porcentagens de número de consultas de pré-natal do grupo *outlier* pelo DBSCAN (grupo 1) e não *outlier* (grupo 0).



(a) Boxplot de nascidos vivos do sexo feminino.

(b) Boxplot de nascidos vivos com anomalia congênita.

Figura 21 – Boxplot para as porcentagens de sexo feminino e de anomalia congênita do grupo *outlier* pelo DBSCAN (grupo 1) e não *outlier* (grupo 0).

3.3.4 Agrupamento Espectral

O algoritmo descrito na Seção 2.2.4 foi implementado pelo código disponível na Seção 6.9. Para a seleção do número de grupos K, foi selecionado o ponto em que os autovalores ordenados da matriz laplaciana mostraram queda importante.

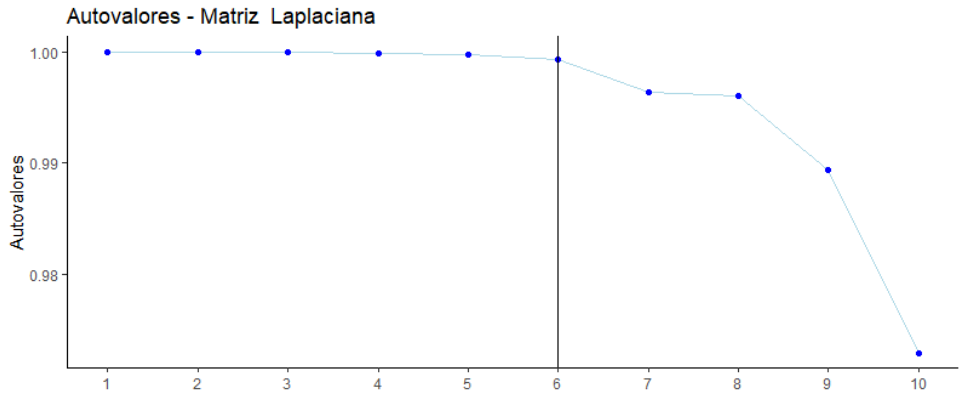


Figura 22 – Autovalores ordenados da Matriz Laplaciana calculada no agrupamento espectral.

Como observado na Figura 22, do sexto para o sétimo autovalor temos uma queda considerável de valor, o que mostra um $K = 6$ ótimo. Assim, realizando o agrupamento com o parâmetro K selecionado, o resultado obtido está na Tabela 12.

Grupo	1	2	3	4	5	6
n	12	17	5	5524	3	9

Tabela 12 – Número de observações por *cluster* do agrupamento espectral.

3.4 Seleção do Melhor Método

Os métodos propostos na Seção 2.3 para avaliação da qualidade dos agrupamentos são considerados para a comparação dos métodos ajustados na Seção 3.3 e para a seleção do método mais adequado para os dados de indicadores obstétricos municipais. Os índices de Davies-Bouldin, Silhueta e Calinski-Harabasz foram implementados utilizando o pacote `clusterSim` (Walesiak e Dudek, 2020), já o índice de Dunn foi obtido pelo pacote `clValid` (Brock et al., 2008).

Na Tabela 13 temos em verde os 5 melhores (ou 6 em casos de empate) resultados para cada métrica calculada, de acordo com sua regra de interpretação.

Por essa tabela vemos que os agrupamentos feitos pelo método hierárquico aglomerativo do vizinho mais próximo (*Single*) se mostraram melhores para os dois índices DB calculados, tanto considerando o centróide como valor referencial quanto o medóide, obtendo os melhores valores para os menores números de grupos. Ainda, esse método também é o melhor avaliado pelo índice de Dunn (D). Porém, analisando os grupos obtidos por esses agrupamentos (Tabela 7), vemos que eles resultaram, exceto quando $K = 7$ e $K = 8$, em um grupo com a grande maioria dos municípios e o restante tendo observações únicas.

Método		Métrica de Validação					
		Davies-Bouldin		Dunn	Silhueta	Calinski-Harabasz	
		Centróide	Medóide			Centróide	Medóide
Agrupamentos por Particionamento	k-médias	2,17	2,84	0,01	0,09	437,34	349,14
	PAM	2,42	2,77	0,01	0,06	399,98	387,07
	CLARA	1,99	2,32	0,01	0,11	343,26	309,67
Agrupamento por Densidade	DBSCAN	0,70	0,76	0,15	0,28	41,51	39,31
Agrupamento Espectral	Algoritmo NJW	1,22	1,43	0,13	0,60	153,16	139,68
Agrupamentos Hierárquicos	BIRCH	1,11	1,14	0,05	0,45	204,84	189,67
	diana com $K = 3$	0,91	0,91	0,08	0,70	282,97	270,37
	diana com $K = 4$	0,72	0,72	0,08	0,70	204,15	194,77
	diana com $K = 5$	0,80	0,76	0,09	0,61	205,23	194,53
	diana com $K = 6$	0,81	0,90	0,09	0,61	172,78	162,36
	diana com $K = 7$	0,99	1,20	0,06	0,44	207,61	185,54
	diana com $K = 8$	0,91	1,09	0,07	0,44	181,80	163,36
	Centróide com $K = 3$	0,27	0,31	0,21	0,79	111,11	108,44
	Centróide com $K = 4$	0,25	0,28	0,21	0,77	84,49	82,43
	Centróide com $K = 5$	0,25	0,27	0,21	0,69	67,65	66,18
	Centróide com $K = 6$	0,39	0,45	0,14	0,69	68,60	65,77
	Centróide com $K = 7$	0,40	0,47	0,15	0,69	70,92	67,94
	Centróide com $K = 8$	0,39	0,46	0,15	0,65	62,88	60,28
	Ward com $K = 3$	2,50	3,46	0,02	0,18	365,77	290,19
	Ward com $K = 4$	2,53	2,63	0,02	0,06	376,49	332,17
	Ward com $K = 5$	2,38	2,54	0,02	0,07	379,82	333,93
	Ward com $K = 6$	2,19	2,56	0,02	0,07	372,25	320,81
	Ward com $K = 7$	2,14	2,53	0,02	0,06	358,69	300,67
	Ward com $K = 8$	2,08	2,71	0,02	0,06	350,02	288,29
	Complete com $K = 3$	0,62	0,63	0,07	0,72	250,23	239,52
	Complete com $K = 4$	0,46	0,47	0,07	0,70	183,10	173,92
	Complete com $K = 5$	0,54	0,62	0,09	0,67	174,28	165,41
	Complete com $K = 6$	0,68	0,89	0,10	0,67	162,53	151,88
	Complete com $K = 7$	0,78	0,92	0,11	0,61	142,54	133,28
	Complete com $K = 8$	0,69	0,76	0,11	0,61	126,28	118,68
	Average com $K = 3$	0,27	0,31	0,21	0,79	111,11	108,44
	Average com $K = 4$	0,25	0,28	0,21	0,77	85,10	83,29
	Average com $K = 5$	0,23	0,26	0,22	0,76	71,72	70,15
	Average com $K = 6$	0,29	0,32	0,15	0,74	115,25	111,30
	Average com $K = 7$	0,29	0,31	0,15	0,69	99,21	95,97
	Average com $K = 8$	0,44	0,49	0,16	0,65	104,87	101,16
	Single com $K = 3$	0,15	0,15	0,33	0,77	69,35	68,05
	Single com $K = 4$	0,16	0,16	0,32	0,76	56,35	55,22
	Single com $K = 5$	0,19	0,19	0,27	0,69	46,43	45,66
	Single com $K = 6$	0,18	0,18	0,24	0,69	48,90	48,09
	Single com $K = 7$	0,22	0,25	0,23	0,69	52,71	51,60
	Single com $K = 8$	0,23	0,26	0,23	0,61	46,67	45,61

Tabela 13 – Valores das métricas de validação para cada método implementado.

Na Tabela 14 vemos que os municípios separados do grande grupo do agrupamento que se mostrou melhor pelos índices DB e D são, apenas, valores extremos, o que pode ser verificado

nos Apêndices D e B. Esse resultado mostra a sensibilidade do método à presença de *outliers*, gerando um resultado que não é interessante ao estudo, já que objetivamos encontrar os grupos similares de município brasileiros.

Palmeirândia - MA	Lagoa do Barro do Piauí - PI
Iguaí - BA	André da Rocha - RS
Pedras Altas - RS	Anhanguera - GO

Tabela 14 – Municípios dos grupos unitários formados pelo agrupamento hierárquico do vizinho mais próximo com $K = 8$.

Já o índice de silhueta S indicou como melhores agrupamentos os resultantes dos métodos hierárquicos do centróide e da média das distâncias (*Average*) para $K = 3$ e $K = 4$, que apresentam o mesmo problema do método do vizinho mais próximo, separando observações extremas.

Temos, então, os índices de Calinski-Harabasz (CH) com valor referencial sendo o centróide e medóide indicando os mesmos métodos como os melhores para os dados. Em ambos os métodos de particionamento K-médias e PAM resultaram em maiores valores, sendo o K-médias melhor quando considerado o centróide como valor de referência e o PAM quando o medóide foi utilizado.

Como pode ser visto nas Tabelas 4 e 5, nenhum desses métodos apresentou o problema de separar grupos unitários, o que se mostrou como a principal desvantagens dos métodos hierárquicos, com exceção do método Ward, que acabou resultando em alguns dos melhores agrupamentos de acordo com o índice CH.

Assim, considerando seus valores obtidos nos índices de Calinski-Harabasz, e sua performance ligeiramente melhor que o método PAM nos demais índices, o agrupamento obtido pelo método K-médias pode ser escolhido como o mais adequado para os dados em estudo.

3.5 Análise de Resultado

A partir do agrupamento pelo método K-médias selecionado na Seção 3.4, podemos analisar os grupos formados a respeito das variáveis obstétricas do estudo. Nas Tabelas 15 e 16 temos as medidas resumo dos indicadores por grupo formado no agrupamento selecionado. O número de municípios em cada um dos 7 clusters é dado por: $n_1 = 573$, $n_2 = 1587$, $n_3 = 1387$, $n_4 = 936$, $n_5 = 404$, $n_6 = 206$ e $n_7 = 477$.

Prematuridade					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	7,39	3,49	0,00	7,37	19,35
2	12,48	3,09	2,78	12,09	32,00
3	11,05	2,99	0,00	10,84	25,00
4	7,90	3,20	0,00	8,10	20,00
5	17,01	5,63	0,00	16,51	62,95
6	10,99	5,56	0,00	10,66	50,00
7	11,70	3,84	2,17	11,28	37,50
Gestação Múltipla					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	0,86	1,45	0,00	0,00	10,53
2	2,33	1,48	0,00	2,34	8,33
3	1,80	1,41	0,00	1,79	9,52
4	1,05	1,48	0,00	0,00	14,29
5	6,63	3,60	0,00	6,13	25,00
6	1,67	2,21	0,00	0,00	10,53
7	1,40	1,04	0,00	1,38	7,84
Parto Cesárea					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	64,06	14,70	23,30	64,29	100,00
2	70,33	11,91	33,83	70,49	100,00
3	47,60	11,58	9,06	47,02	94,03
4	65,19	15,16	12,03	64,91	100,00
5	66,59	15,11	27,27	67,74	100,00
6	63,70	15,48	26,04	65,36	96,55
7	40,43	13,84	5,62	40,83	82,36
Nenhuma Consulta Pré-Natal					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	0,71	1,27	0,00	0,00	9,84
2	0,75	0,96	0,00	0,52	8,88
3	1,07	1,12	0,00	0,87	8,70
4	0,69	1,07	0,00	0,00	9,09
5	0,74	1,39	0,00	0,00	11,43
6	0,79	1,31	0,00	0,00	6,12
7	4,98	8,34	0,00	2,72	91,61

Tabela 15 – Medidas descritivas dos indicadores obstétricos por grupo obtido, sendo elas a média (média), desvio padrão (dp), med (med), mínimo (min) e máximo (max).

7 ou mais Consultas Pré-Natal					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	80,52	10,44	46,67	81,67	100,00
2	81,51	8,16	50,79	82,66	100,00
3	73,83	9,96	43,41	74,16	100,00
4	79,53	10,53	44,44	80,70	100,00
5	79,91	11,71	20,00	81,88	100,00
6	78,74	12,19	33,68	81,08	100,00
7	47,31	12,30	3,73	49,47	80,85
Apgar do 1º Minuto Menor que 7					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	4,23	3,03	0,00	3,94	18,18
2	4,82	2,32	0,00	4,76	16,67
3	8,36	4,78	0,00	7,30	66,02
4	3,70	2,39	0,00	3,57	14,52
5	6,78	4,39	0,00	6,25	37,50
6	5,57	3,95	0,00	5,21	19,05
7	4,96	2,74	0,00	4,58	22,22
Apgar do 5º Minuto Menor que 7					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	0,58	0,87	0,00	0,00	4,55
2	0,76	0,69	0,00	0,72	4,55
3	1,84	1,87	0,00	1,47	30,51
4	0,53	0,75	0,00	0,00	6,67
5	1,47	1,73	0,00	1,05	9,38
6	1,13	1,52	0,00	0,47	9,09
7	1,13	0,89	0,00	1,01	6,67
Anomalia Congênita					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	0,42	0,72	0,00	0,00	4,17
2	0,73	0,69	0,00	0,64	3,08
3	0,78	0,69	0,00	0,70	3,33
4	0,50	0,70	0,00	0,00	4,00
5	0,70	1,12	0,00	0,00	7,14
6	4,60	2,41	2,56	3,85	18,75
7	0,54	0,55	0,00	0,42	3,11
Peso Menor que 2.500g					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	4,93	2,78	0,00	5,20	16,13
2	9,24	2,22	0,00	9,09	21,43
3	8,47	2,44	0,00	8,33	21,74
4	5,90	2,50	0,00	5,97	16,67
5	13,84	4,38	4,55	13,23	37,50
6	8,40	4,56	0,00	8,11	21,74
7	7,44	2,41	0,00	7,30	21,43
Sexo Feminino					
Grupo	**média**	**dp**	**min**	**med**	**max**
1	41,75	5,50	0,00	43,06	50,00
2	48,25	3,77	26,09	48,54	66,67
3	48,77	4,01	26,09	48,73	72,22
4	53,53	4,70	47,02	52,46	83,33
5	49,93	7,23	14,29	49,61	75,00
6	48,92	8,37	27,59	48,16	74,19
7	48,80	3,17	31,17	48,96	59,82

Tabela 16 – Cont.: Medidas descritivas dos indicadores obstétricos por grupo obtido, sendo elas a média (média), desvio padrão (dp), med (med), mínimo (min) e máximo (max).

Como é possível observar na Figura 23 temos como algo bem marcante na distribuição geográfica dos grupos o *cluster* 7, que abrange praticamente toda a região Norte e o estado do Maranhão. Ainda, vemos que os estados das regiões Sul e Sudeste aparentam serem semelhantes quanto aos grupos de municípios.

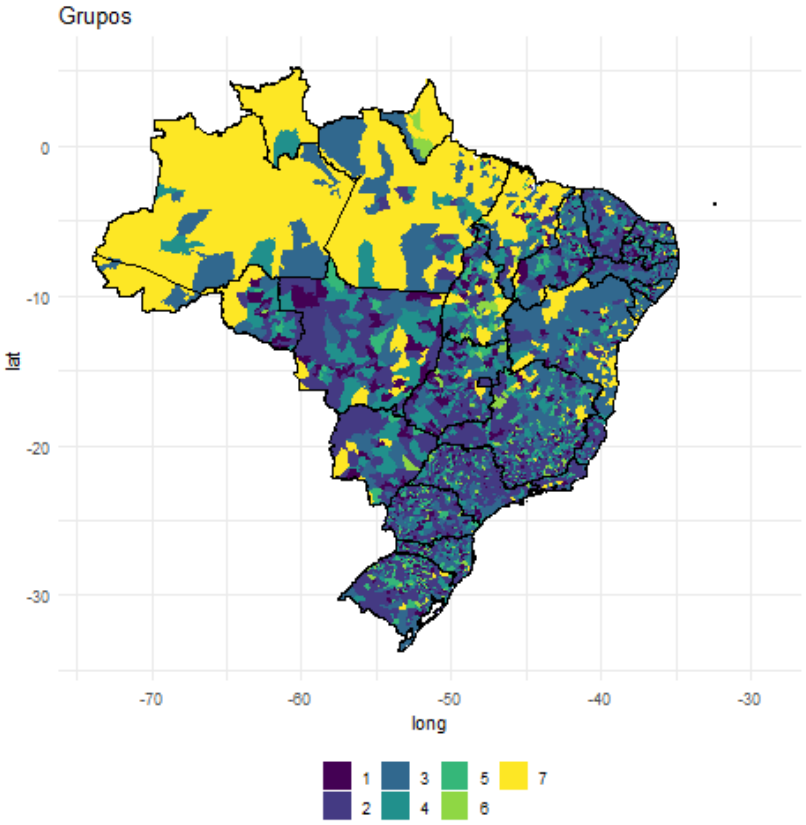


Figura 23 – Mapa dos *clusters* obtidos.

Para o entendimento de como as variáveis utilizadas estão definindo cada um dos grupos formados, temos a árvore de decisão da Figura 24, que foi ajustada pelo pacote `rpart` (Therneau e Atkinson, 2019). Nela é possível verificar as variáveis que se mostram mais importantes para a delimitação dos grupos. Dessa forma, temos que os indicadores obstétricos mais importantes para o agrupamento foram: percentual de parto cesárea, de peso menor que 2.500g, de 7 ou mais consultas de pré-natal, de gestação múltipla, de sexo feminino, de Apgar do 1º minuto menor que 7, de prematuridade e de anomalia congênita.

Temos que, de todas as 10 variáveis utilizadas para a análise, as porcentagens de Apgar do 5º minuto menor que 7 e de nenhuma consulta pré-natal foram as únicas que não se mostraram relevantes. Essas duas, porém, apresentam correlação moderada com Apgar do 1º minuto menor que 7 e com 7 ou mais consultas de pré-natal (Figura 13), respectivamente, além de serem diretamente relacionadas pela própria natureza das variáveis.

A partir da árvore de decisão montada temos as seguintes interpretações:

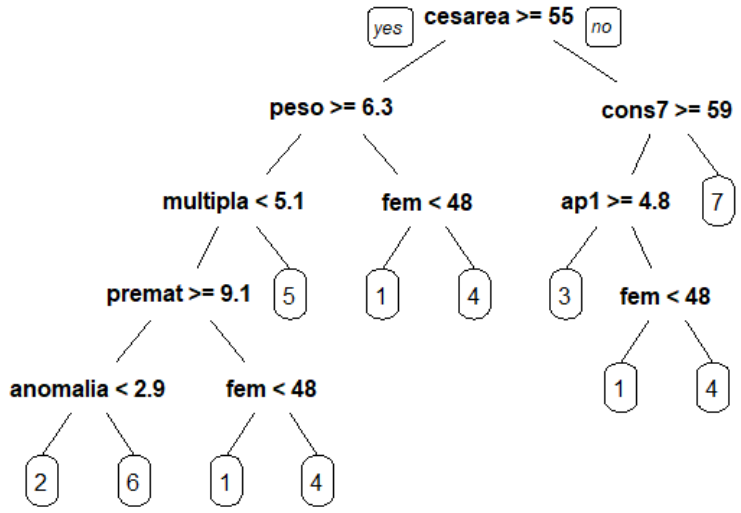


Figura 24 – Árvore de decisão dos grupos.

- Os grupos 1 e 4 se diferenciam pelo percentual válido de nascidos vivos do sexo feminino, em que o grupo 4 apresenta municípios com porcentagens maiores que 48%, enquanto o grupo 1 possui maior taxa igual a 50%.
- O grupo 3 se caracteriza por apresentar menores porcentagens de parto cesárea, maiores porcentagens de nascidos vivos com 7 consultas de pré natal ou mais (maior de 50%) e de Apgar do primeiro minuto menores que 7.
- O grupo 5 é caracterizado por apresentar porcentagem de cesárea maior que 55%, maiores porcentagens de peso menor que 2.500g (maiores que 6,3% dos nascidos vivos) e maiores porcentagens de gestação múltipla (maiores que 5,1%).
- Os grupos 2 e 6 são caracterizados por apresentarem as maiores porcentagens de prematuridade (maior que 9,1%), mas são diferenciados pelo fato do grupo 6 apresentar as maiores porcentagens de anomalia congênita (maior que 2,9%).
- O grupo 7 apresenta menores porcentagens de parto cesárea e tem os menores percentuais de 7 ou mais consultas pré-natal.

É interessante, também, observamos como as capitais dos estados estão distribuídas entre os grupos. Nesse sentido, temos que as 26 capitais estaduais e o Distrito Federal foram alocados apenas nos grupos 2, 3, 4 e 7. Nossas capitais, então, não estão no grupo que se caracteriza por ter menos de 48% de taxa válida de nascidos vivos do sexo feminino (grupo 1), não se assemelham aos municípios com maiores percentuais de gestação múltipla (grupo 5) e também não estão no grupo com porcentagem de prematuridade maior que 9,1% e com alta porcentagem de anomalia congênita (grupo 6).

Como vemos na Tabela 17, os grupos 2 e 3 são os que mais possuem capitais agrupadas neles, seguidas pelo grupo 7, que tem 5 de suas 7 capitais situadas na Região Norte, enquanto Campo Grande, capital do Mato Grosso do Sul é a única que foi alocada no grupo 4. A lista de capitais estaduais e seus respectivos grupos está no Apêndice E.

Grupo	1	2	3	4	5	6	7
número de capitais	0	10	10	1	0	0	6

Tabela 17 – Número de capitais por grupo.

Na Figura 25 podemos observar que os grupos 2, 3 e 7 são os de maior número de nascidos vivos, sendo que o grupo 7 se destaca com valores maiores. Enquanto os grupos 1, 5 e 6 reúne municípios com menores número de nascidos vivos.

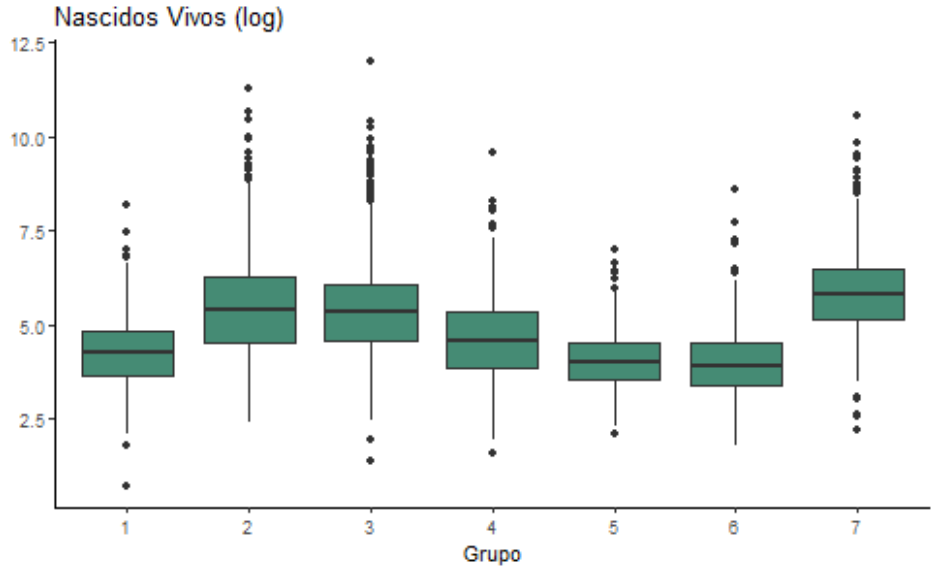


Figura 25 – Gráficos de boxplot do número de nascidos vivos por município por grupo.

Como análise complementar, vamos analisar como as variáveis socioeconômicas se comportam na partição proposta, lembrando que os indicadores obtidos são referentes ao Censo do ano de 2010 do IBGE. Assim, na Figura 26 podemos visualizar que o grupo 7 se destaca com os maiores valores de fecundidade total e índice de Gini, enquanto apresenta baixos índices de desenvolvimento humano e renda *per capita*. Podemos observar, também, que os municípios do grupo 3 apresentam essas mesmas tendências, porém, com uma diferença menos acentuada em relação aos demais grupos. Já o grupo 2 apresenta, no geral, melhores indicadores de IDHM e renda *per capita*.

Dessa forma, podemos ver que, além do menores índices de 7 ou mais consultas de pré-natal, os municípios do grupo 7 também apresentam piores indicadores socioeconômicos, que, como podemos ver no mapa da Figura 23, estão concentrados nas região norte do país.

Enquanto isso, temos as capitais estaduais divididas entre grupos de melhores (grupo 2) e piores (grupo 3) indicadores de renda e desenvolvimento humano.

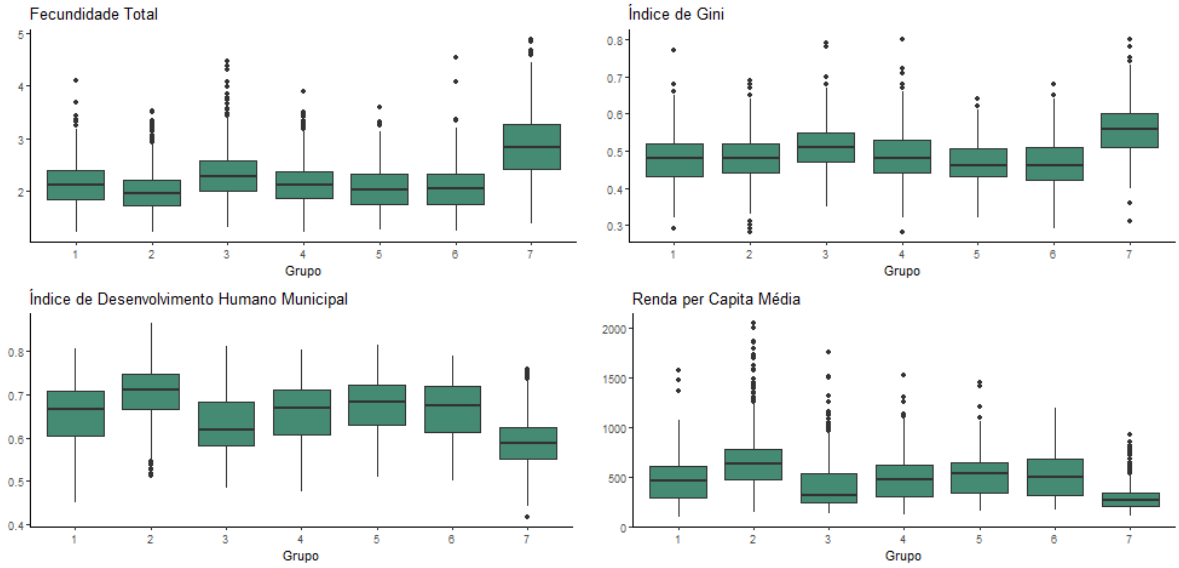


Figura 26 – Gráficos de boxplot das variáveis socioeconômicas por grupo.

4 Estudo de simulação

Tendo em vista os conteúdos teóricos apresentados no Capítulo 2 e a aplicação discutida no Capítulo 3, nesse capítulo será apresentado um estudo de simulação, com o intuito de aprofundar a discussão acerca dos métodos de agrupamento e também das métricas de validação consideradas.

Uma vez que para a aplicação não é possível definir de antemão quantos *clusters* devem haver no conjunto de dados, é interessante considerar uma situação em que a definição de grupo é mais clara.

Na Seção 4.1 será apresentada uma maneira de criar grupos, de tal forma que estes possam ser identificados a partir de um conjunto de variáveis definidas no intervalo (0, 1), assim como foi feito na aplicação. Em seguida, na Seção 4.2 serão apresentados os principais resultados deste estudo de simulação e uma discussão sobre a relação destes com a teoria apresentada, assim como as conclusões obtidas no capítulo anterior.

4.1 Cenários para o estudo

Levando em consideração a aplicação ilustrada no Capítulo 3, em que as variáveis utilizadas para o agrupamento são todas proporções, ou seja, valores entre o intervalo (0, 1), nesse estudo de simulação serão utilizadas variáveis com distribuição somente nesse intervalo. Dessa forma, uma boa escolha para gerar valores para o estudo de simulação é a distribuição beta. Para facilitar a interpretação dos valores gerados, será considerada a seguinte função de densidade

$$f(x;\mu,\phi) = \frac{\Gamma(\phi)}{\Gamma(\mu\phi)\Gamma((1-\mu)\phi)} x^{\mu\phi-1}(1-x)^{(1-\mu)\phi-1}, \quad 0 < x < 1,$$

em que $E(X) = \mu$ e $\text{Var}(X) = (\mu(1-\mu))/(1+\phi)$, tal que o parâmetro μ define o valor esperado de X, enquanto que ϕ pode ser visto como um parâmetro de precisão. Quanto maior o valor de ϕ menor será o valor da variância de X.

Para considerar uma situação similar àquela discutida no capítulo anterior, para o estudo de simulação serão considerados 7 diferentes grupos. Com o intuito de utilizar valores de médias próximos daqueles considerados na aplicação, o seguinte vetor de médias foi utilizado para gerar 7 diferentes variáveis com distribuição beta com diferentes médias

$$\mu = (0,01,\ 0,02,\ 0,05,\ 0,10,\ 0.40,\ 0.60,\ 0.75)'. \tag{10}$$

Esses valores serão utilizados como base na geração de cada variável X_i, porém estes podem ser alterados para diferenciar um determinado grupo. Para todas as distribuições foi considerado o

valor de $\phi = 150$. Os valores dentro do vetor (10) podem ser denotados por μ_{x_i} para identificar as 7 diferentes médias das variáveis X_i, $i = 1, \ldots, 7$.

Em um primeiro cenário, para que essas variáveis pudessem identificar os diferentes grupos, foi utilizado um parâmetro para separar a distribuição de probabilidade de um respectivo grupo. Esse parâmetro β_j é utilizado da seguinte maneira para separar a média dos grupos

$$\mu_i = \frac{\exp(\log(\mu_{x_i}/(1-\mu_{x_i})) + \beta_j)}{1 + \exp(\log(\mu_{x_i}/(1-\mu_{x_i})) + \beta_j)}, \tag{11}$$

em que se $i \neq j$, então $\beta_j = 0$; caso contrário β_j é o j-ésimo termo do vetor

$$\beta = (1, 1, 1, 1, 1, -1, -1).$$

Para exemplificar, note que $\mu_{x_1} = 0.01$, logo para observações do grupo 1, a média $\mu_1 = \exp(\log(0,01/0,99) + 1)/(1 + \exp(\log(0,01/0,99) + 1)) = 0,0267$. Para os grupos 2 a 7, essa média μ_1 é igual a 0,01. Essa média μ_i é utilizada para gerar os valores da variável X_i. Dessa maneira, a variável X_1 "identifica" o grupo 1, a variável X_2 "identifica" o grupo 2 e assim por diante. Uma ilustração para uma particular amostra gerada desse cenário é apresentada na Figura 27, considerando as 7 variáveis geradas e os 7 grupos de interesse.

Um segundo cenário foi considerado para obter uma situação em que mais de uma variável seja necessária para "identificar" um grupo. Nesse segundo cenário, com exceção da variável X_1, para todas as variáveis as médias definidas em (10) serão alteradas por dois grupos. Para ajudar no entendimento de como esses parâmetros vão afetar as médias dos diferentes grupos e das diferentes variáveis é apresentada a seguinte matriz B, com elementos b_{ij}.

$$B = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

As colunas estão vinculadas às variáveis X_i, enquanto que as linhas estão associadas aos grupos de 1 a 7. A forma como esses valores vão alterar os valores das médias vai ser feito da mesma maneira que em (11). Por exemplo, para a variável X_1 somente para o grupo 1 é que há um parâmetro $b_{11} = 1$ alterando o valor da média. Para a váriavel X_2 tanto para valores do grupo 1, quanto para valores do grupo 2, os parâmetros $b_{12} = b_{22} = 1$ irão afetar a média de X_2 da mesma forma. Logo, considerando somente a variável X_2 não seria possível diferenciar o grupo 1 do grupo 2. De maneira similar, a média da variável X_3 também é influenciada por observações de dois grupos: 2 e 3. Uma particular amostra gerada a partir desse segundo cenário é apresentada na Figura 28.

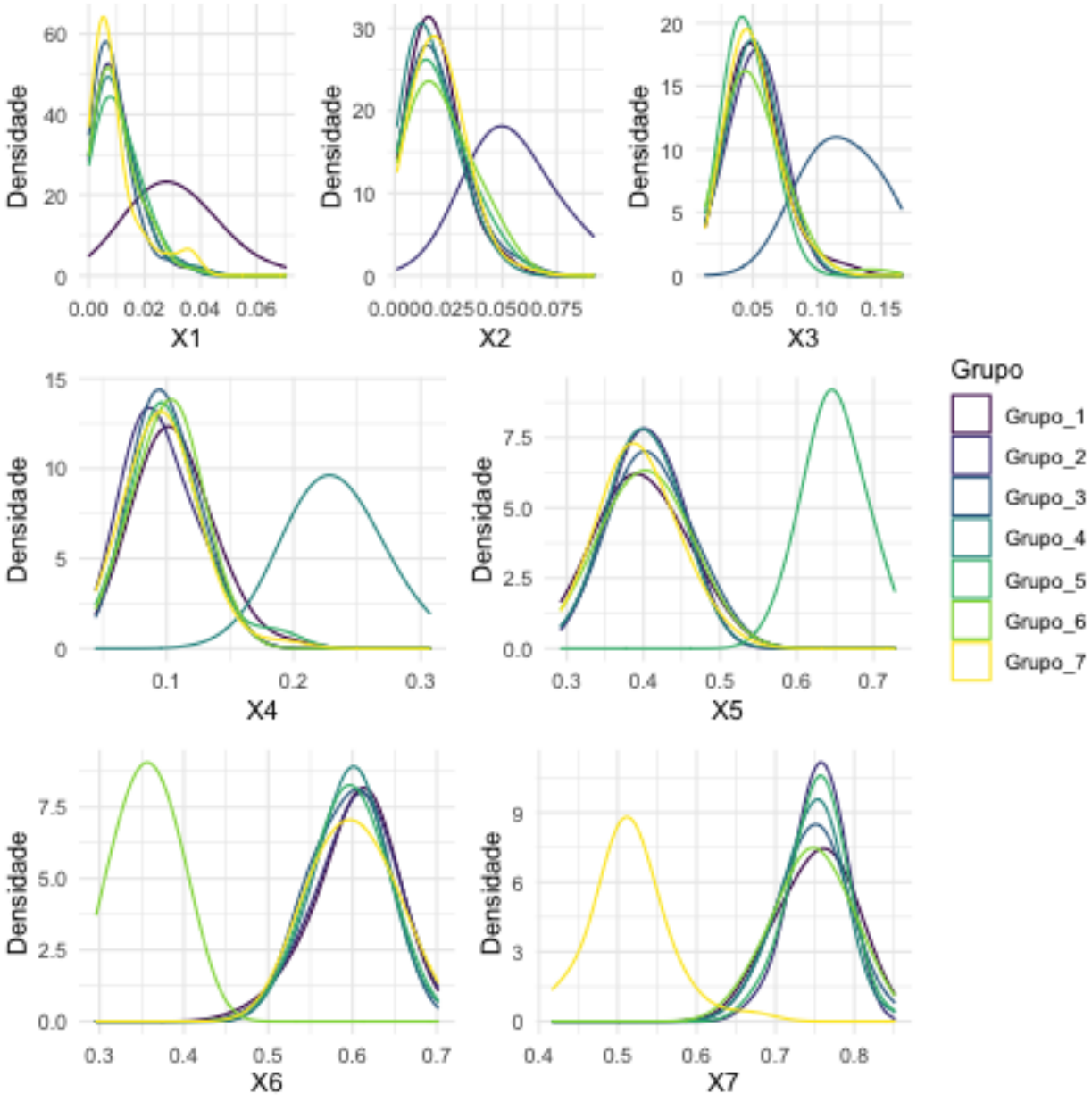


Figura 27 – Densidades estimadas para uma particular amostra gerada para os diferentes grupos a partir das 7 variáveis consideradas, $X_1, \ldots, X_7$, considerando o primeiro cenário para o estudo de simulação.

Além de considerar esses dois cenários para gerar os dados, também serão considerados diferentes tamanhos para cada grupo: 50, 100 e 200. Também serão adicionadas variáveis com distribuição beta, porém sem nenhuma relação com a identificação dos grupos. Essas variáveis serão chamadas de variáveis de ruído e serão consideradas três situações: sem variáveis de ruídos, 3 e 20 variáveis de ruído.

Para esse estudo de simulação não serão considerados os métodos de agrupamento espectral e DBSCAN. O primeiro por motivo de tempo computacional, o qual é relativamente

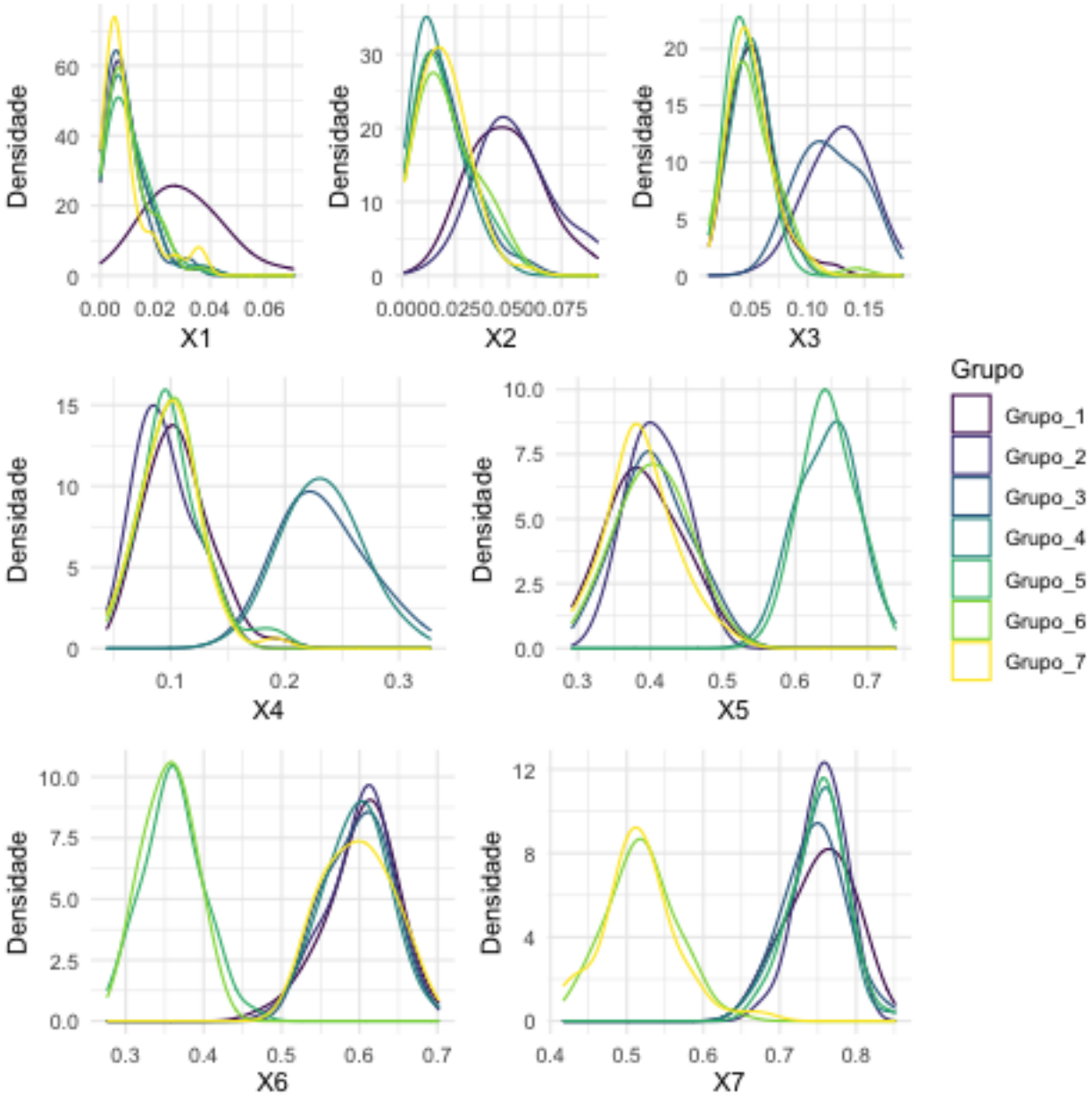


Figura 28 – Densidades estimadas para uma particular amostra gerada para os diferentes grupos a partir das 7 variáveis consideradas, $X_1, \ldots, X_7$, considerando o segundo cenário para o estudo de simulação.

maior comparado com os outros métodos e tendo em vista que cada cenário do estudo será repetido 1.000 vezes. O segundo por motivo de sensibilidade dos seus hiperparâmetros (ϵ e MinPts), os quais devem ser escolhidos de maneira individual para cada aplicação e afetam os resultados do agrupamento de forma considerável.

Para todos os cenários serão computados os diferentes índices de validação do agrupamento, assim como o tempo computacional. Além disso, será feita uma comparação para alguns métodos quando o total de grupos é especificado de forma incorreta.

4.2 Resultados e discussão

Para cada cenário de simulação foram obtidas 1.000 réplicas do banco de dados gerados, segundo as duas maneiras apresentadas na Seção 4.1. Cada cenário do estudo de simulação é composto então por uma combinação dos seguintes fatores:

- Geração dos dados: "Cenário 1"e "Cenário 2", apresentados na Seção 4.1.
- Tamanho de cada grupo: $n_k = 50$, 100 e 200, para $k = 1, \ldots, 7$.
- Variáveis de ruído: "Sem variáveis de ruído", "3 variáveis de ruído"e "20 variáveis de ruído".

4.2.1 Tempo computacional

Com relação ao tempo computacional, foi possível perceber que os métodos BIRCH e DIANA apresentarem tempos médios (em segundos) muito maiores que os demais métodos. Os resultados podem ser vistos na Figura 29. O método PAM também parece estar em um patamar acima dos demais métodos. Para facilitar a comparação, a Figura 30 mostra os mesmos resultados, porém retirando os métodos BIRCH, DIANA e PAM.

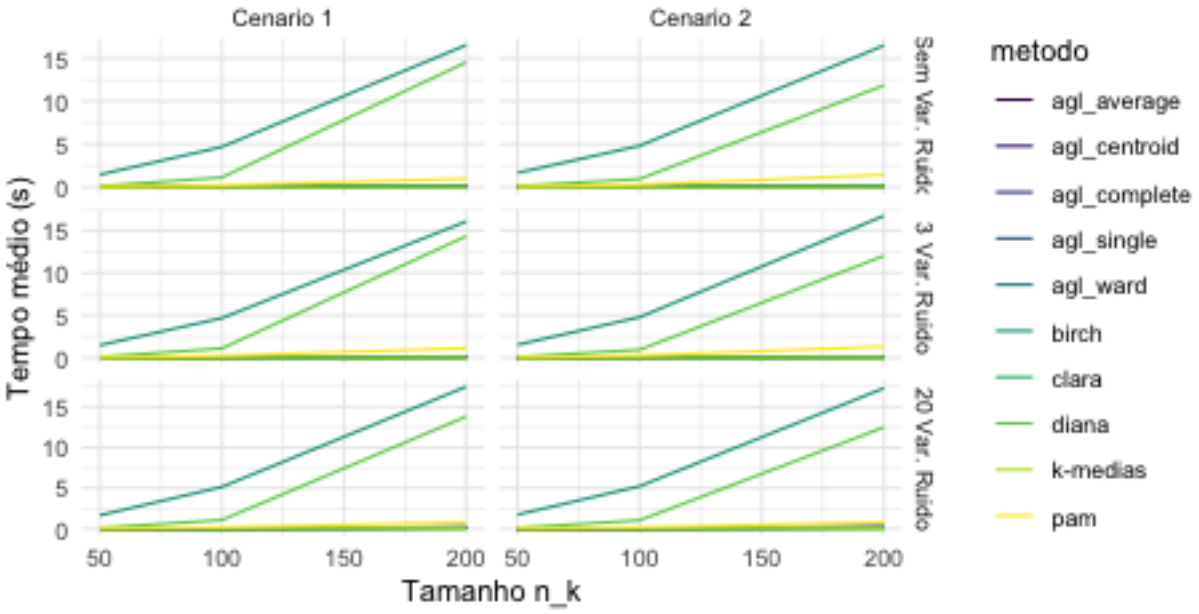


Figura 29 – Tempo computacional médio (em segundos) para os diferentes cenários do estudo de simulação.

É possível visualizar então que para a maioria dos métodos parece haver um aumento significativo no tempo médio conforme se aumenta os tamanhos dos grupos. Uma exceção desse resultado é vista somente para os métodos K-médias e CLARA.

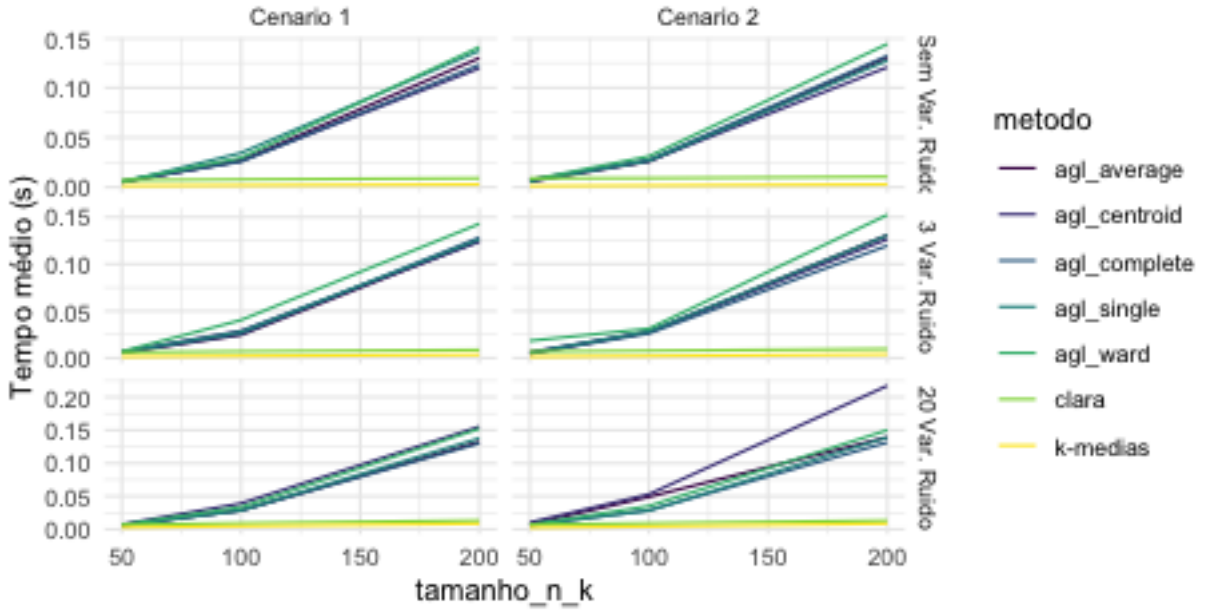


Figura 30 – Tempo computacional médio (em segundos) para os diferentes cenários do estudo de simulação, retirando os métodos BIRCH, DIANA e PAM.

Com respeito aos diferentes cenários propostos, não é possível ver grandes diferenças considerando o tempo computacional. Além disso, para a maioria dos métodos parece haver um aumento do tempo computacional médio quando se aumentam o número de variáveis de ruído.

4.2.2 Métricas de validação

Conforme as métricas de validação apresentadas na Seção 2.3, aqui nessa parte do estudo serão comparadas as métricas para os diferentes métodos de agrupamento, considerando que o número de grupos é corretamente identificado. Nesse caso, para todos os métodos será fixado o número de *clusters* igual a 7. Em seguida, na próxima subseção será feita uma comparação variando esse número de grupos para alguns métodos selecionados.

Primeiramente, nas Figuras 31 e 32 são apresentados os valores do índice Davies-Bouldin, considerando as duas formas diferentes para definir os pontos centrais para o cálculo de distância: centróides e medóides, respectivamente. É importante lembrar que para essa métrica quanto menor o valor, melhor é o resultado do agrupamento.

Considerando essa métrica, para a maioria dos cenários o método hierárquico considerando o método dos vizinhos mais próximos parece ser aquele com melhor desempenho. Outro método hierárquico que tem performance similar é o método do centróide. E em seguida também outro agrupamento hierárquico que considera o método da média das distâncias. É importante notar que para esse primeiro índice, os métodos hierárquicos parecem ter o melhor desempenho.

Com relação aos diferentes cenários propostos, não parece haver uma diferença muito

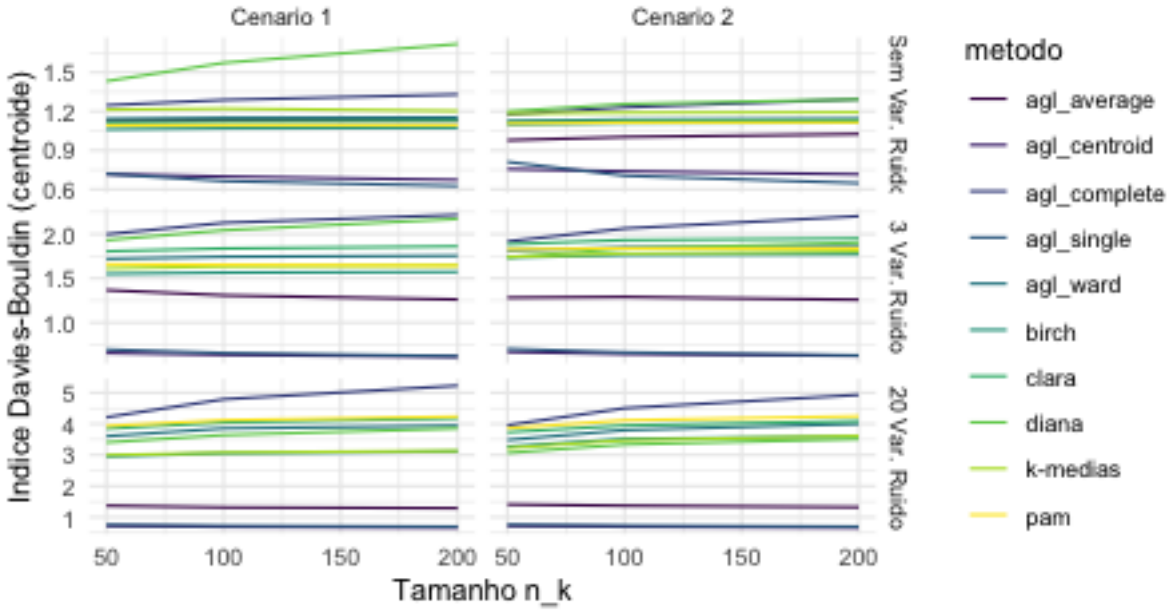


Figura 31 – Média do índice Davies-Bouldin para os diferentes cenários do estudo de simulação, considerando centróides como pontos centrais para o cálculo de distâncias.

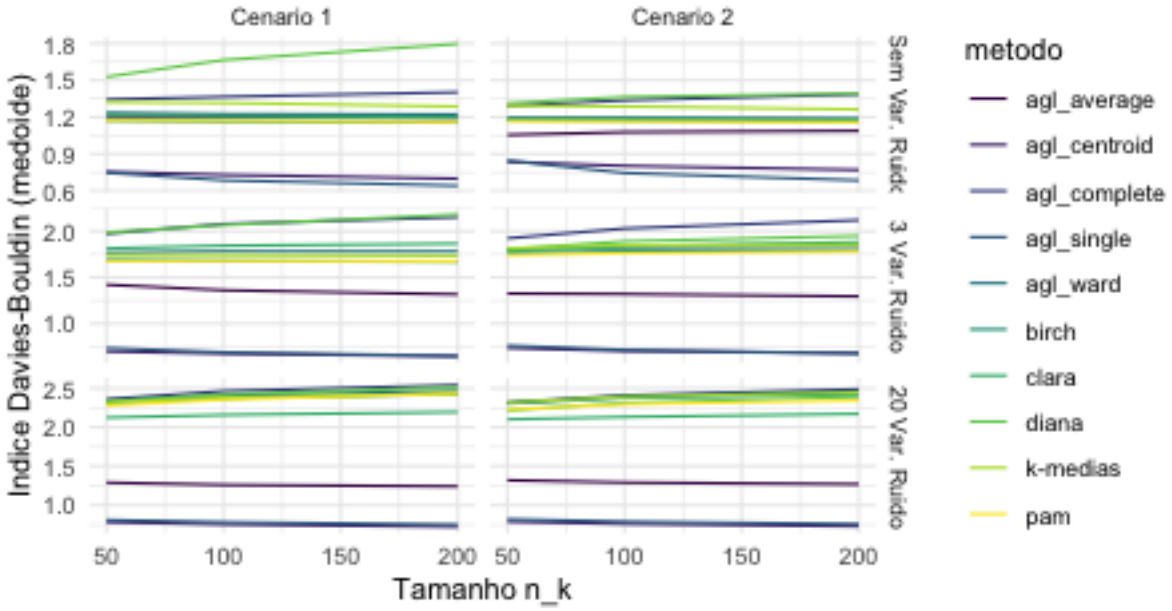


Figura 32 – Média do índice Davies-Bouldin para os diferentes cenários do estudo de simulação, considerando medóides como pontos centrais para o cálculo de distâncias.

grande quando é feita a comparação entre os cenários 1 e 2, mantendo o mesmo número de variáveis de ruído. O mesmo pode ser dito com relação quando se aumenta o número de variáveis de ruído para o mesmo cenário de geração dos dados.

Em seguida, os resultados obtidos considerando o índice Dunn são apresentados na Figura 33. Para essa métrica, maiores valores indicam um melhor resultado do agrupamento,

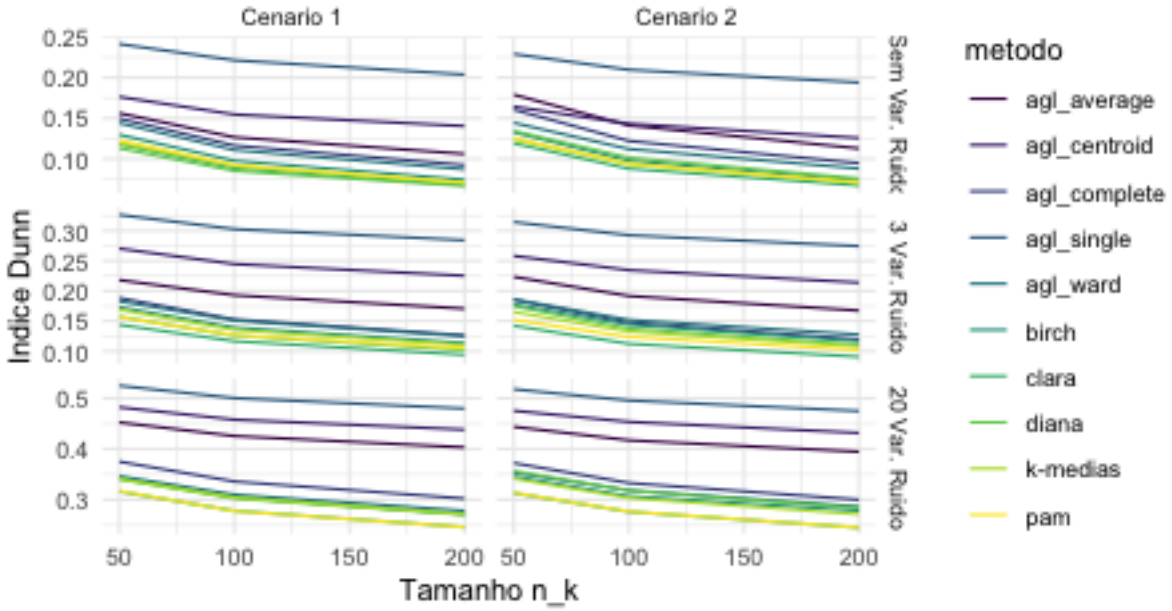


Figura 33 – Média do índice Dunn para os diferentes cenários do estudo de simulação.

portanto novamente o método que deveria ser escolhido como melhor seria o método do vizinho mais próximo, método hierárquico, assim como o índice anterior.

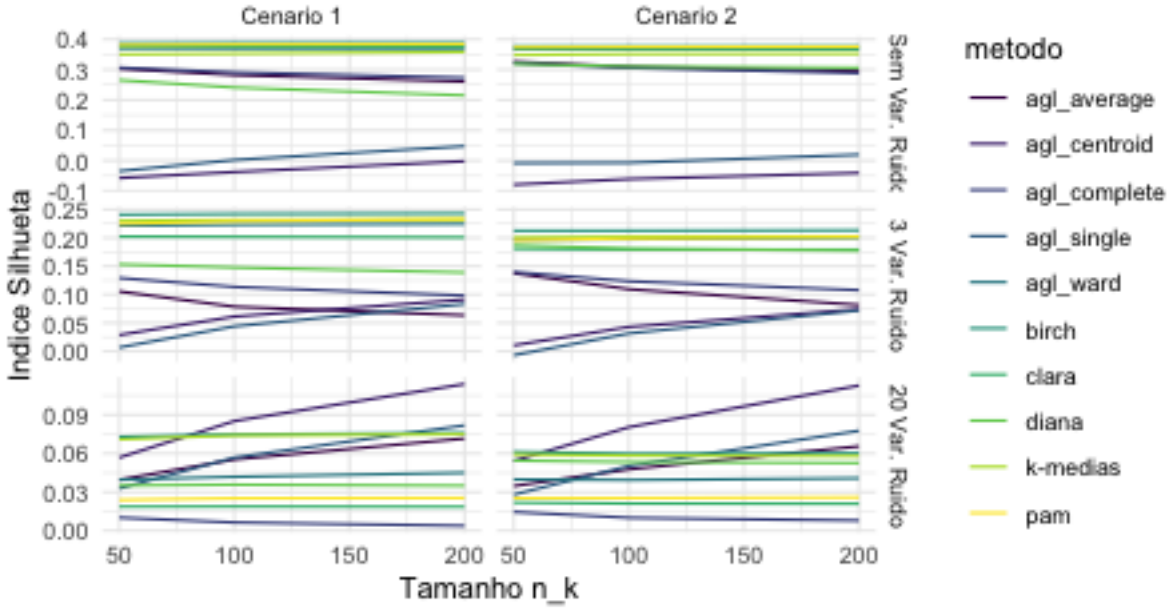


Figura 34 – Média do índice Silhueta para os diferentes cenários do estudo de simulação.

Para a métrica da silhueta, em que maiores valores indicam melhor resultado do agrupamento, parece haver uma maior variação das conclusões segundo esse índice. Os resultados estão dispostos na Figura 34. Por exemplo, para as situações em que não há variáveis de ruído, parece haver um grande número de métodos que retornam os maiores valores segundo essa métrica. Além disso, para os casos com 20 variáveis de ruído, a conclusão parece mudar quando

o tamanho dos grupos aumenta. Por exemplo, para o cenário 1 quando n_k é igual a 50, os métodos BIRCH e K-médias apresentam os melhores resultados. Quando o tamanho do grupo é igual a 200, os algoritmos com as melhores performances são o método hierárquico do centróide e o método hierárquico dos vizinhos mais próximos.

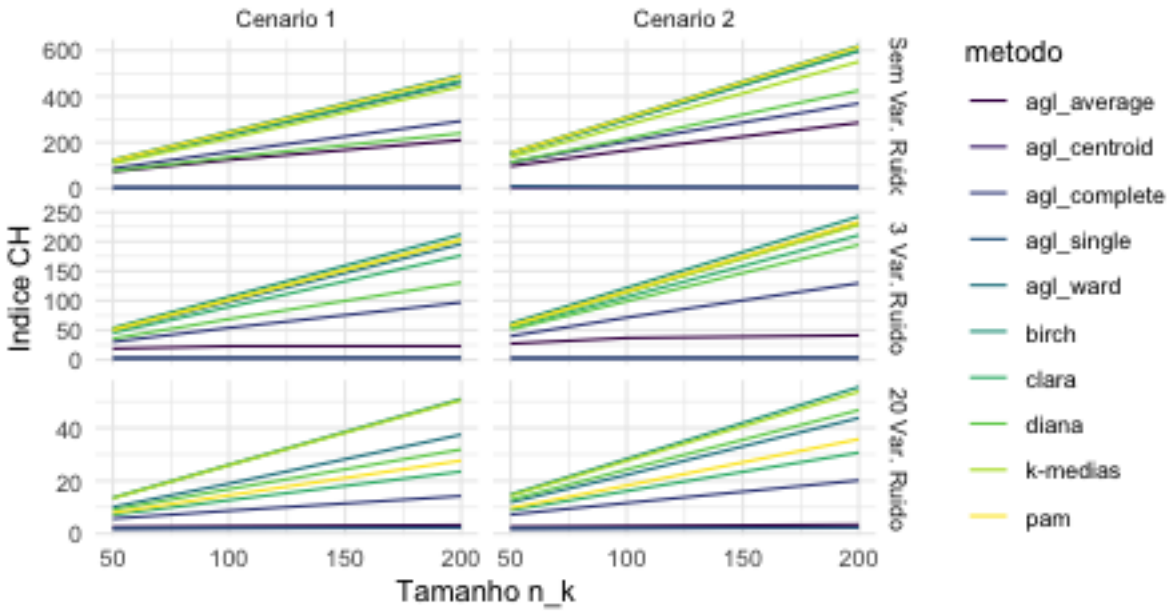


Figura 35 – Média do índice Calinski-Harabasz para os diferentes cenários do estudo de simulação, considerando pontos centrais como centróides.

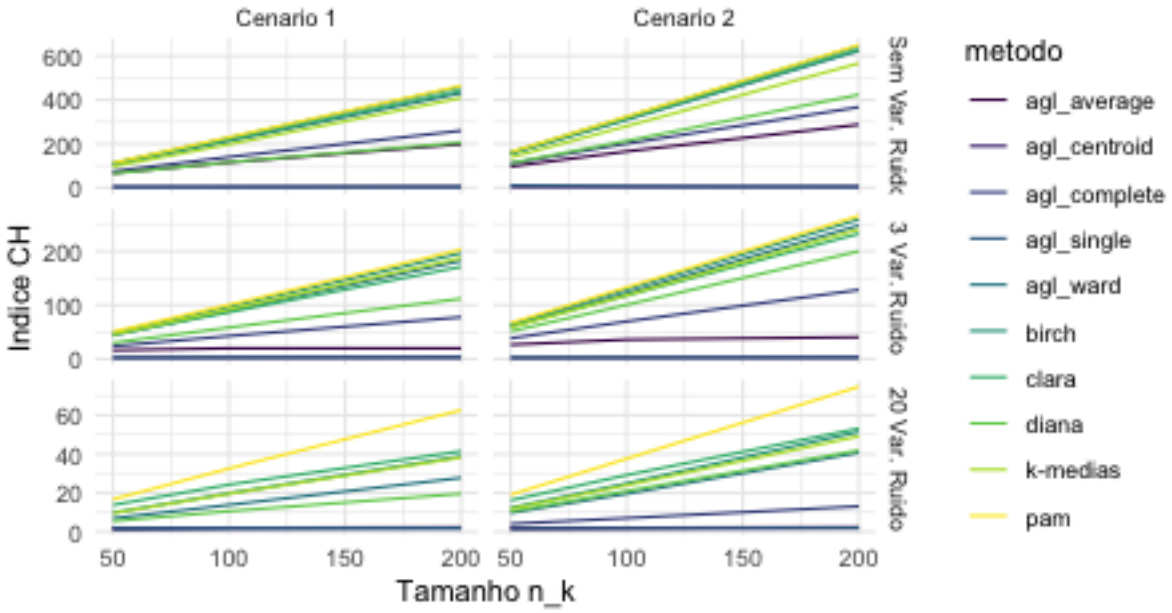


Figura 36 – Média do índice Calinski-Harabasz para os diferentes cenários do estudo de simulação, considerando pontos centrais como medóides.

Para o cálculo do índice Calinski-Harabasz foram considerados dois métodos implemen-

tados no software R, considerando pontos centrais como centróides ou medóides. Os resultados estão apresentados nas Figuras 35 e 36, respectivamente. Para essa métrica os melhores resultados foram observados pelos métodos BIRCH e K-médias, para a maioria das situações. Em particular, para o caso com 20 variáveis de ruído e considerando os pontos centrais como medóides, o método PAM se destacou como o melhor método, para ambos cenários de geração dos dados. Também é possível notar que os valores médios observados aumentam conforme se aumenta o tamanho dos grupos.

É importante identificar nessa parte do estudo de simulação que a depender da métrica utilizada, diferentes métodos de agrupamento teriam sido apontados como aqueles com os melhores resultados. Essa discussão é essencial tendo em vista os resultados observados no capítulo anterior. Em particular não se observou grande influência no tamanho dos grupos, com exceção do índice Silhueta. Adicionalmente, os dois cenários criados para geração dos dados não apresentaram grandes diferenças em suas conclusões.

4.2.3 Comparação da performance para diferentes números de *clusters*

Na subseção anterior, para todos as comparações feitas entre os diferentes métodos de agrupamento foi considerado que se conhecia o valor exato de agrupamentos, i.e., 7 grupos, conforme explicação na Seção 4.1. Um outro ponto importante a ser investigado aqui nesse estudo de simulação é analisar como esses métodos se comportam quando são considerados diferentes valores para o número de *clusters*.

Para facilitar essa comparação foram selecionados 5 métodos de agrupamento somente: os métodos hierárquicos dos centróides, dos vizinho mais próximos e dos vizinhos mais distantes, DIANA e K-médias. Considerando os resultados observados na subseção anterior, foi utilizado somente o tamanho de grupos igual a 50.

A Figura 37 mostra o resultado da média de quatro índices apresentados anteriormente, para as diferentes combinações das situações. Antes de comentar os resultados, é importante apontar que o resultado ideal nesses gráficos seria observar um "V" invertido para aquelas métricas em que maiores valores indicam o melhor ajuste, com ponto de máximo no número de *clusters* igual a 7, pois esse é o valor considerado como correto. De maneira análoga, para os índices em que menores valores indicam melhor performance, seria ideal observar um formato de "V" com mínimo valor quando o número de *clusters* escolhidos fosse igual a 7. No Apêndice F foram colocados os gráficos boxplot para cada uma das situações analisadas, em que é possível observar de maneira mais completa a variação dos índices para os diferentes métodos de agrupamento.

Analisando os valores médios da Figura 37 é possível notar que o método K-médias é aquele que apresenta o comportamento com o formato de "V" ou "V" invertido no maior número de situações. Nessa parte do estudo de simulações é possível perceber uma maior dificuldade dos

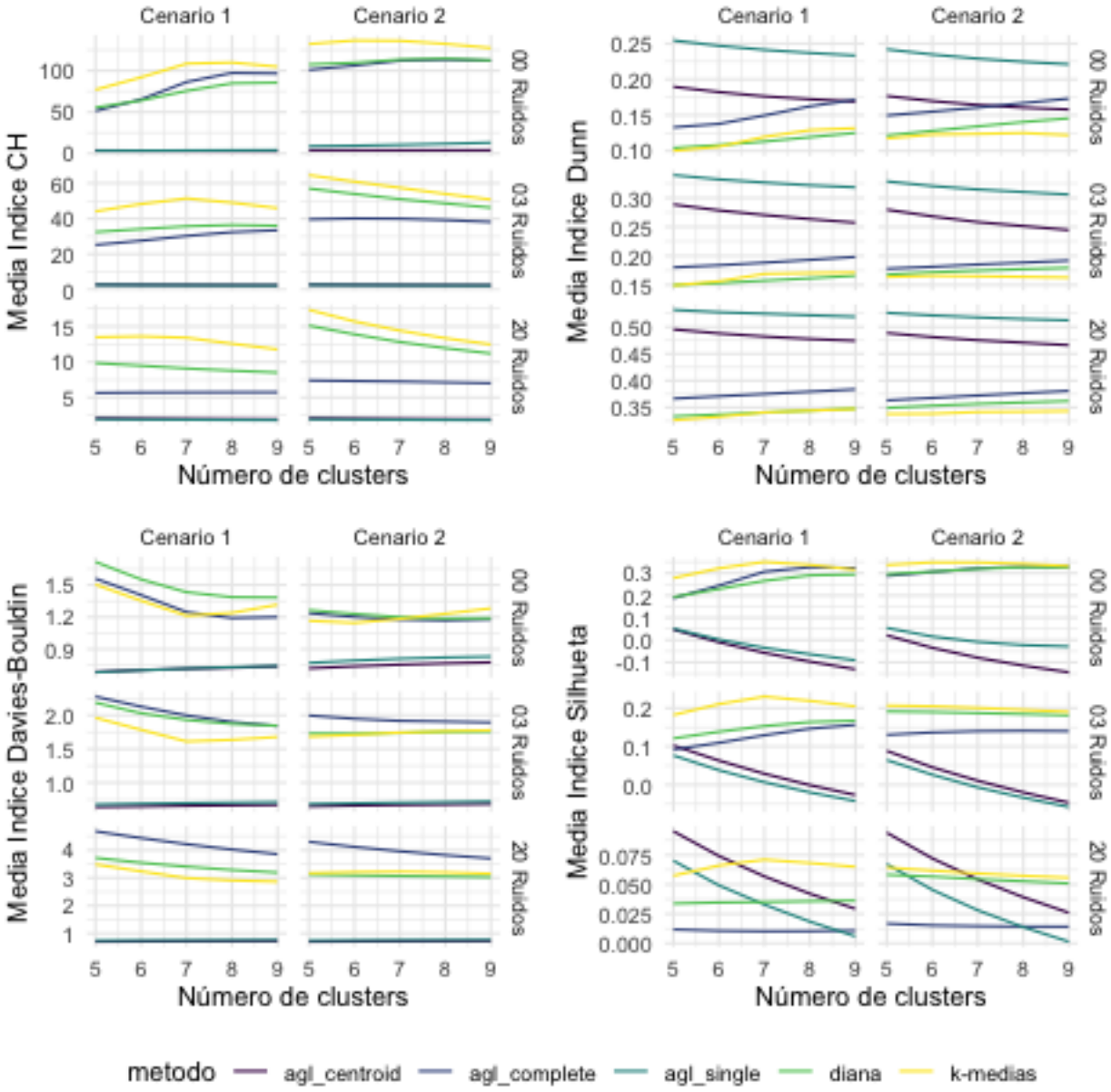


Figura 37 – Comparação dos valores médios das diferentes métricas de validação para os cinco métodos de agrupamento escolhidos, quando o número de clusters para cada método varia de 5 a 9.

métodos de agrupamento quando são comparados os cenários 1 e 2 de geração de dados. Em especial, quando há variáveis de ruído, todos os métodos de agrupamento parecem ter dificuldade de identificar que o número de *clusters* deve ser igual a 7.

É importante também apontar alguns comportamentos observados dentre os métodos de agrupamento e as diferentes métricas. Por exemplo, utilizando o índice silhueta, o método dos centróides e do vizinho mais próximo parecem ser mais conservadores, selecionando sempre um número de *clusters* menor que 7. Tal comportamento é notável em todas as situações analisadas.

Assim como na análise anterior desse estudo de simulação, a depender da métrica utilizada um método de agrupamento diferente seria selecionado como o melhor. Novamente apenas considerando o índice silhueta é que as conclusões poderiam ser diferentes quando considerados diferentes números de *clusters*. Porém, aqui tendo em vista a comparação entre os diferentes números de grupos escolhidos, é possível dizer que o método K-médias apresentou os melhores resultados.

5 Conclusão

Esse trabalho apresentou algumas das técnicas existentes para a resolução do problema de agrupamento no Capítulo 2, a partir de diferentes métodos de agrupamento por particionamento e métodos hierárquicos, além de abordar também o agrupamento por densidade DBSCAN e o método espectral. Nesse mesmo capítulo foram apresentados 4 índices de avaliação de agrupamento, que têm por objetivo indicar a qualidade de um agrupamento realizado, em que, com cálculos próprios, avaliam se o particionamento indicado está com grupos com baixa variância interna e bem separados .

A respeito das técnicas de agrupamento por particionamento, que a partir de um número K pré determinado particionam os dados utilizando medidas de dissimilaridade, foram apresentados 3 diferentes métodos, sendo eles o K-médias (Seção 2.2.1.1), que busca minimizar a variância total intragrupo considerando como valor de referência para cada iteração do algoritmo um ponto médio (o centróide) e o K-medóides (Seção 2.2.1.2), que se difere do K-médias por considerar observações do conjunto de dados como valores de referência para a minimização da variância intragrupo. Para o k-medóide foi apresentado o algoritmo PAM, que apresenta um maior custo computacional, e o CLARA, que surge como uma solução para o problema de tempo computacional.

Para os métodos hierárquicos foram apresentadas as abordagens aglomerativa e divisiva (Seção 2.2.2), em que para a aglomerativa foram considerados 5 diferentes medidas de dissimilaridade entre grupos, enquanto para o método divisivo consideramos o algoritmo DIANA. Outro método que utiliza técnicas hierárquicas foi apresentado, o BIRCH, que realiza primeiramente um agrupamento de forma hierárquica a partir de vetores de informação de cada grupo, para depois realizar outro tipo de agrupamento com os dados já pré-agrupados.

Já na Seção 2.2.3 apresentamos o método de agrupamento por densidade DBSCAN, que particiona os dados de forma a localizar regiões do espaço que contem menos observações, a partir de um número mínimo de pontos e um parâmetro de raio a ser considerado, sendo assim, capaz de identificar *outliers* no conjunto de dados.

Por fim foi desenvolvido o método de agrupamento espectral (Seção 2.2.4), que visa primeiramente reduzir a dimensionalidade dos dados e, então, agrupar esse novo conjunto obtido, utilizando nesse trabalho o algoritmo de Ng, Jordan e Weiss.

Para a avaliação da qualidade dos agrupamentos foram apresentados, na Seção 2.3, quatro diferentes índices utilizados: Davies-Bouldin, Dunn, Silhueta e Calinski-Harabasz. Índices esses que, por diferentes cálculos, tentam medir a variância dentro dos *clusters* e a dissimilaridade entre eles, de forma a obter uma relação entre esses valores que permita a interpretação do quão bem separados e densos estão os grupos formados.

No Capítulo 3 foi apresentado o conjunto de dados contendo os 10 indicadores considerados de saúde obstétrica dos municípios brasileiros, obtidos através do SINASC, para a aplicação da metodologia estudada. Na Seção 3.2 foi feita uma análise exploratória dos dados que permitiu observar a existência de valores preocupantes a respeito de alguns indicadores, tais como o número de consultas de pré-natal e partos cesáreas, além de identificar municípios com taxas discrepantes dos demais em certas variáveis.

Na Seção 3.3 foram apresentados os resultados dos agrupamentos obtidos por cada método. Para os três método de particionamento utilizados (Seção 3.3.1), foi identificado o mesmo número ótimo de grupos para os dados. Para os métodos hierárquicos (Seção 3.3.2) pudemos ver que em sua maioria eles foram sensíveis aos ruídos existentes nos dados, mesmo para diferentes valores de K, apresentando resultados pouco informativos, o que demonstra uma limitação desses métodos. Já os métodos aglomerativo de Ward e divisivo DIANA se mostraram mais robustos a esses dados, fornecendo resultados mais interessantes ao estudo. O método BIRCH apresentou comportamento semelhante aos demais hierárquicos em sua primeira etapa, com vários subgrupos contendo poucas observações.

O método DBSCAN foi implementado na Seção 3.3.3, em que pudemos ver que mesmo sendo capaz de detectar *outliers*, ainda realizou o agrupamento de poucas observações em seus grupos. Foi realizada, então, uma análise dos municípios identificados como *outliers*, onde concluiu-se que eram observações que tinham baixo número de nascidos vivos e maiores porcentagens de prematuridade, gestação múltipla, peso menor que 2.500g, anomalias congênita e valores de apgar menores que 7, ou seja, municípios com piores indicadores. Para o agrupamento espectral na Seção 3.3.4 obtivemos, mais uma vez, grupos com poucas observações.

Para a seleção do melhor método (Seção 3.4) foram utilizados os índices estudados, considerando medóide e centróide como valores de referência para os índices de Davies-Bouldin e Calinski-Harabasz. Pudemos ver duas tendências, uma sendo com os índices de Davies-Bouldin (DB), Dunn (D) e Silhueta (S) indicando os hierárquicos como melhores grupos, e outra com os de particionamento e o hierárquico de Ward sendo melhores avaliados pelo índice de Calinski-Harabasz (CH). O que pode ser explicado pela forma como cada índice é obtido, em que para o CH a medida de separação entre grupos é ponderada pelo seu tamanho, o que favoreceu os agrupamentos com *clusters* com mais observações, enquanto as demais métricas punem mais o pouco afastamento entre grupos, o que acaba por favorecer os agrupamentos que só separaram essas medidas discrepantes, presentes nos nossos dados.

Dessa forma, foi selecionado o agrupamento obtido pelo método K-médias, que foi analisado na Seção 3.5. Em relação às variáveis obstétricas utilizadas no agrupamento, pudemos observar quais se mostraram importantes para a separação feita, em que apenas a indicadora do apgar do quinto minuto menor que 7 e a de nenhuma consulta pré-natal não foram selecionadas.

Tivemos, então, como grupos que mais chamaram a atenção o grupo 7, que concentra principalmente os municípios do norte do país e tem menores percentuais de nascidos vivos

com 7 ou mais consultas pré-natal, além de menores taxas de partos cesáreas. Outros grupos interessantes foram os grupos 2 e 6, que apresentam maiores taxas de prematuridade.

Cruzando, então, os agrupamentos obtidos com alguns indicadores socioeconômicos obtidos com base no Censo de 2010 do IBGE, vemos que o grupo 7 também apresentou os menores índices de desenvolvimento humano municipal e menores valores de renda *per capita* média.

Por fim, foi realizado um estudo de simulação no Capítulo 4, em que foram gerados dados provenientes de 7 grupos por construção, considerando distribuições betas com diferentes parâmetros, em que alguns dos métodos aplicados aos dados obstétricos foram implementados nesses dados simulados. A respeito do tempo computacional, três métodos se mostraram mais custosos, dois sabidamente mais demorados, o DIANA e o PAM, e o BIRCH, que se propõe a ser mais rápido. Esse comportamento do BIRCH pode ser causado pelo algoritmo utilizado, que carece de validação. Ainda sobre tempo computacional, os métodos K-médias e CLARA se mostraram pouco sensíveis ao tamanho dos grupos.

A respeito das medidas de validação, para os índices DB e D os métodos hierárquicos têm melhor desempenho, assim como observado na aplicação aos dados de indicadores obstétricos. Enquanto para o índice S a variação dos cenários afeta na escolha do melhor método, variando entre o BIRCH, K-médias e hierárquicos. Já para o índice CH temos BIRCH e K-médias como melhores para a maioria dos cenários.

Para realizar a comparação para diferentes números de grupos selecionados foram utilizados os métodos hierárquicos aglomerativos dos centróides, dos vizinho mais próximos e dos vizinhos mais distantes, DIANA e K-médias. Nessa avaliação o método K-médias se mostrou o mais adequado, com melhores índices para o número de agrupamento correto, o que corrobora com o trabalho desenvolvido e os resultados obtidos na aplicação das variáveis obstétricas municipais.

6 Códigos

6.1 Dados

```r
### carregando pacotes necess rios
loadlibrary <- function(x) {
  if (!require(x, character.only = TRUE)) {
    install.packages(x, dependencies = T)
    if (!require(x, character.only = TRUE))
      stop("Package not found")
  }
}
packages <- c(
  "readxl",
  "janitor",
  "dplyr",
  "tidyverse",
  "skimr",
  "forcats",
  "stringr",
  "lubridate",
  "readr",
  "ggplot2",
  "summarytools",
  "factoextra",
  "psych",
  "gridExtra",
  "Rcpp"
)
lapply(packages, loadlibrary)

### carregando dados
dados <- readRDS("dados_2019.rds")

### limpando dados
municipios_ignorados <- dados %>%
  filter(str_detect(municipio, "MUNICIPIO IGNORADO"))

dados <- dados %>%
  filter(!municipio %in% municipios_ignorados$municipio) %>%
  select(
    uf,
    municipio,
    codigo,
```

```
    nascidos_vivos,
    starts_with("porc")
  ) %>%
  mutate(
    uf = as.factor(uf),
    codigo = as.factor(codigo),
    porc_peso_menor_2500 = ifelse(is.na(porc_peso_menor_2500) == TRUE,
                                  0,
                                  porc_peso_menor_2500)
  )


## dados normalizados
dados_norm <- as.data.frame(scale(dados[,-c(1:4)]))

## matriz de dissimilaridade
dados_diss <- dist(scale(dados[, -c(1:4)]), method = "euclidean")
```

6.2 Análise Descritiva

```
### tabela resumo
descritiva <-
  round(psych::describe(dados[-c(1:3)]), 2)

### correlograma
corrplot::corrplot(
  cor(dados[-c(1:3)], method = "spearman"),
  type = "lower",
  method = "color",
  addCoef.col = "black",
  tl.col = "darkblue",
  diag = FALSE
)
############################ Graficos de Dispersao
############################## peso X prematuridade
pesoXpremat <-
  ggplot(dados, aes(porc_peso_menor_2500, porc_premat)) + theme_classic() +
  labs(y = "Prematuridade (%)",
       x = "Peso Menor de 2.500g (%)",
       title = "") +
  geom_point(color = "aquamarine4") +
  geom_text(
    data = subset(dados, porc_premat > 34 |
                    porc_peso_menor_2500 > 24 |
                    porc_peso_menor_2500 < 10 &
                    porc_premat > 30 |
                    porc_peso_menor_2500 > 20 &
```

```
                    porc_premat < 10),
    aes(porc_peso_menor_2500,
        porc_premat,
        label = paste0(municipio, ", ", uf)),
    size = 3,
    check_overlap = TRUE
  )

############################ multipla X cesarea
multiplaXcesarea <-
  ggplot(dados, aes(porc_gesta_multipla, porc_cesarea)) + theme_classic() +
  labs(y = "Parto Ces rea (%)",
       x = "Gesta  o M ltipla (%)",
       title = "") +
  geom_point(color = "aquamarine4") +
  geom_text(
    data = subset(
      dados,
      porc_gesta_multipla > 17 |
        porc_gesta_multipla > 10 &
        porc_cesarea < 52.5 |
        porc_cesarea < 10
    ),
    aes(porc_gesta_multipla,
        porc_cesarea,
        label = paste0(municipio, ", ", uf)),
    size = 3,
    check_overlap = TRUE
  )

############################ 0 X 7+ consultas
consultas <-
  ggplot(dados, aes(porc_0_consulta, porc_7mais_consulta)) + theme_classic
    () +
  labs(y = "7 ou mais Consultas Pr -Natal (%)",
       x = "Nenhuma Consulta Pr -Natal (%)",
       title = "") +
  geom_point(color = "aquamarine4") +
  geom_text(
    data = subset(
      dados,
      porc_0_consulta > 20 |
      porc_7mais_consulta < 18
    ),
    aes(porc_0_consulta,
        porc_7mais_consulta,
        label = paste0(municipio, ", ", uf)),
```

```
    size = 3,
    check_overlap = TRUE
  )


############################ APGAR 1 X 5
apgar <-
  ggplot(dados, aes(porc_apgar1_menor_7, porc_apgar5_menor_7)) +
  theme_classic() +
  labs(y = "Apgar do 5   Minuto menor que 7 (%)",
       x = "Apgar do 1   Minuto menor que 7 (%)",
       title = "") +
  geom_point(color = "aquamarine4") +
  geom_text(
    data = subset(
      dados,
      porc_apgar1_menor_7 > 25|
      porc_apgar5_menor_7 > 10
    ),
    aes(porc_apgar1_menor_7,
        porc_apgar5_menor_7,
        label = paste0(municipio, ", ", uf)),
    size = 3,
    check_overlap = TRUE
  )


############################ anomalia X feminino
anomaliaXfem <-
  ggplot(dados, aes(porc_anomalia, porc_fem)) +
  theme_classic() +
  labs(y = "Sexo Feminino (%)",
       x = "Anomalia Cong nita (%)",
       title = "") +
  geom_point(color = "aquamarine4") +
  geom_text(
    data = subset(
      dados,
      porc_anomalia > 7|
        porc_fem < 21 |
        porc_fem > 73
    ),
    aes(porc_anomalia,
        porc_fem,
        label = paste0(municipio, ", ", uf)),
    size = 3,
    check_overlap = TRUE
```

```
  )
```

6.3 Criação dos Mapas

```
############ criando base de dados dos mapas

library(rgdal)
library(leaflet)
library(highcharter)
library(mapview)
library(htmlwidgets)
library(geobr)

### base de dados para constru  o dos mapas
caminho_mapas <-
  "br_municipios_20200807"

mapa_brasil <- readOGR(dsn = caminho_mapas,
                       layer = 'BR_Municipios_2019',
                       verbose = FALSE)

mapa_brasil@data$codigo <-
  substring(mapa_brasil@data$CD_MUN, 1, nchar(mapa_brasil@data$CD_MUN) -
              1) %>%
  as.factor()

### cruzando dados dos mapas com a base de dados no estudo
mapa_dados <- mapa_brasil
mapa_dados@data <- mapa_brasil@data %>%
  left_join(dados[,-c(1:2)])

## criando vari vel identificadora do munic pio
mapa_dados@data$id = rownames(mapa_dados@data)

## criando base de dados municipais
mapa_dados_pontos = ggplot2::fortify(mapa_dados, region = "id")

mapa_dados_df = dplyr::inner_join(mapa_dados_pontos,
                                  mapa_dados@data,
                                  by = "id")

### gerando data frame para UF
caminho_uf <-
  "estados_2010/estados_2010"

mapa_uf <- readOGR(dsn = caminho_uf,
                   layer = 'estados_2010',
```

```
                    verbose = FALSE)

## ajustando formata  o dos dados
mapa_uf@data$nome <-
  iconv(mapa_uf@data$nome, from = 'UTF-8', to = 'ASCII//TRANSLIT')

mapa_uf@data$uf <- mapa_uf@data$sigla %>%
  as.factor()

## criando base de dados estaduais
mapa_uf_pontos = ggplot2::fortify(mapa_uf, region = "id")

mapa_uf_df = dplyr::inner_join(mapa_uf_pontos,
                               mapa_uf@data,
                               by = "id")


################################ mapas
memory.limit(size = 100000)# impossibilidade de alocar vetor por tamanho
mapa_nascidos_vivos <- ggplot(mapa_dados_df %>%
                                filter(codigo %in% dados$codigo)) +
  theme_minimal() +
  geom_polygon(aes(
    x = long,
    y = lat,
    group = group,
    fill = log(nascidos_vivos)
  )) +
  coord_equal() +
  theme_minimal() +
  scale_fill_viridis_c("") +
  theme(legend.position = 'bottom') +
  ggtitle("Nascidos Vivos (log)") +
  geom_path(
    aes(long, lat, group = group),
    color = "black",
    size = 0.05,
    data = mapa_uf_df
  )

memory.limit(size = 100000)
mapa_porc_premat <- ggplot(mapa_dados_df %>%
                             filter(codigo %in% dados$codigo)) +
  theme_minimal() +
  geom_polygon(aes(
    x = long,
    y = lat,
```

```
    group = group,
    fill = porc_premat
  )) +
  coord_equal() +
  theme_minimal() +
  scale_fill_viridis_c("") +
  theme(legend.position = 'bottom') +
  ggtitle("Prematuridade (%)") +
  geom_path(
    aes(long, lat, group = group),
    color = "black",
    size = 0.05,
    data = mapa_uf_df
  )

memory.limit(size = 100000)
mapa_porc_gesta_multipla <- ggplot(mapa_dados_df %>%
                                     filter(codigo %in% dados$codigo)) +
  theme_minimal() +
  geom_polygon(aes(
    x = long,
    y = lat,
    group = group,
    fill = porc_gesta_multipla
  )) +
  coord_equal() +
  theme_minimal() +
  scale_fill_viridis_c("") +
  theme(legend.position = 'bottom') +
  ggtitle("Gesta  o M ltipla (%)") +
  geom_path(
    aes(long, lat, group = group),
    color = "black",
    size = 0.05,
    data = mapa_uf_df
  )

memory.limit(size = 100000)
mapa_porc_cesarea <- ggplot(mapa_dados_df %>%
                              filter(codigo %in% dados$codigo)) +
  theme_minimal() +
  geom_polygon(aes(
    x = long,
    y = lat,
    group = group,
    fill = porc_cesarea
  )) +
```

```
138   coord_equal() +
139   theme_minimal() +
140   scale_fill_viridis_c("") +
141   theme(legend.position = 'bottom') +
142   ggtitle("Parto Ces rea (%)") +
143   geom_path(
144     aes(long, lat, group = group),
145     color = "black",
146     size = 0.05,
147     data = mapa_uf_df
148   )
149
150 memory.limit(size = 100000)
151 mapa_porc_0_consulta <- ggplot(mapa_dados_df %>%
152                                  filter(codigo %in% dados$codigo)) +
153   theme_minimal() +
154   geom_polygon(aes(
155     x = long,
156     y = lat,
157     group = group,
158     fill = porc_0_consulta
159   )) +
160   coord_equal() +
161   theme_minimal() +
162   scale_fill_viridis_c("") +
163   theme(legend.position = 'bottom') +
164   ggtitle("Nenhuma Consulta Pr -Natal (%)") +
165   geom_path(
166     aes(long, lat, group = group),
167     color = "black",
168     size = 0.05,
169     data = mapa_uf_df
170   )
171
172 memory.limit(size = 100000)
173 mapa_porc_7mais_consulta <- ggplot(mapa_dados_df %>%
174                                      filter(codigo %in% dados$codigo)) +
175   theme_minimal() +
176   geom_polygon(aes(
177     x = long,
178     y = lat,
179     group = group,
180     fill = porc_7mais_consulta
181   )) +
182   coord_equal() +
183   theme_minimal() +
184   scale_fill_viridis_c("") +
```

```
185   theme(legend.position = 'bottom') +
186   ggtitle("7 ou mais Consultas Pr  -Natal (%)") +
187   geom_path(
188     aes(long, lat, group = group),
189     color = "black",
190     size = 0.05,
191     data = mapa_uf_df
192   )
193
194 memory.limit(size = 100000)
195 mapa_porc_apgar1_menor_7 <- ggplot(mapa_dados_df %>%
196                                      filter(codigo %in% dados$codigo)) +
197   theme_minimal() +
198   geom_polygon(aes(
199     x = long,
200     y = lat,
201     group = group,
202     fill = porc_apgar1_menor_7
203   )) +
204   coord_equal() +
205   theme_minimal() +
206   scale_fill_viridis_c("") +
207   theme(legend.position = 'bottom') +
208   ggtitle("Apgar do 1   minuto menor que 7 (%)") +
209   geom_path(
210     aes(long, lat, group = group),
211     color = "black",
212     size = 0.05,
213     data = mapa_uf_df
214   )
215
216 memory.limit(size = 100000)
217 mapa_porc_apgar5_menor_7 <- ggplot(mapa_dados_df %>%
218                                      filter(codigo %in% dados$codigo)) +
219   theme_minimal() +
220   geom_polygon(aes(
221     x = long,
222     y = lat,
223     group = group,
224     fill = porc_apgar5_menor_7
225   )) +
226   coord_equal() +
227   theme_minimal() +
228   scale_fill_viridis_c("") +
229   theme(legend.position = 'bottom') +
230   ggtitle("Apgar do 5   minuto menor que 7 (%)") +
231   geom_path(
```

```
232     aes(long, lat, group = group),
233     color = "black",
234     size = 0.05,
235     data = mapa_uf_df
236   )
237
238 memory.limit(size = 100000)
239 mapa_porc_anomalia <- ggplot(mapa_dados_df %>%
240                                filter(codigo %in% dados$codigo)) +
241   theme_minimal() +
242   geom_polygon(aes(
243     x = long,
244     y = lat,
245     group = group,
246     fill = porc_anomalia
247   )) +
248   coord_equal() +
249   theme_minimal() +
250   scale_fill_viridis_c("") +
251   theme(legend.position = 'bottom') +
252   ggtitle("Anomalia Cong nita (%)") +
253   geom_path(
254     aes(long, lat, group = group),
255     color = "black",
256     size = 0.05,
257     data = mapa_uf_df
258   )
259
260 memory.limit(size = 100000)
261 mapa_porc_peso_menor_2500 <- ggplot(mapa_dados_df %>%
262                                       filter(codigo %in% dados$codigo)) +
263   theme_minimal() +
264   geom_polygon(aes(
265     x = long,
266     y = lat,
267     group = group,
268     fill = porc_peso_menor_2500
269   )) +
270   coord_equal() +
271   theme_minimal() +
272   scale_fill_viridis_c("") +
273   theme(legend.position = 'bottom') +
274   ggtitle("Peso menor que 2.500g  (%)") +
275   geom_path(
276     aes(long, lat, group = group),
277     color = "black",
278     size = 0.05,
```

```
    data = mapa_uf_df
  )

memory.limit(size = 100000)
mapa_porc_fem <- ggplot(mapa_dados_df %>%
                          filter(codigo %in% dados$codigo)) +
  theme_minimal() +
  geom_polygon(aes(
    x = long,
    y = lat,
    group = group,
    fill = porc_fem
  )) +
  coord_equal() +
  theme_minimal() +
  scale_fill_viridis_c("") +
  theme(legend.position = 'bottom' ) +
  ggtitle("Sexo Feminino (%)") +
  geom_path(
    aes(long, lat, group = group),
    color = "black",
    size = 0.05,
    data = mapa_uf_df
  )
```

6.4 Métodos de Particionamento

```
####################### K medias
## escolhendo k pelo metodo do cotovelo
cotovelo_kmedias <- factoextra::fviz_nbclust(dados_norm,
                                             kmeans,
                                             method = "wss") +
  geom_vline(xintercept = 7, linetype = 2) +
  labs(x = "Numero de Grupos",
       y = "Variancia Total Intragrupo",
       title = "K-medias")

## ajustando k-medias com o numero de grupos escolhido
set.seed(1122)
k_medias <- kmeans(dados_norm,
                   centers = 7)
#saveRDS(k_medias, "k_medias.rds")

####################### K medoids
## escolhendo k pelo metodo do cotovelo
cotovelo_pam <- factoextra::fviz_nbclust(dados_norm,
                                         cluster::pam,
```

```
                                          method = "wss") +
  geom_vline(xintercept = 7, linetype = 2) +
  labs(x = "Numero de Grupos",
       y = "Variancia Total Intragrupo",
       title = "PAM")

cotovelo_clara <- factoextra::fviz_nbclust(dados_norm,
                                           cluster::clara,
                                           method = "wss") +
  geom_vline(xintercept = 7, linetype = 2) +
  labs(x = "Numero de Grupos",
       y = "Variancia Total Intragrupo",
       title = "CLARA")

## ajustando k-medoids com o numero de grupos escolhido
pam <- cluster::pam(dados_norm,
                    k = 7)

clara <- cluster::clara(dados_norm,
                        k = 7,
                        samples = 10)
```

6.5 Métodos Hierárquicos

```
####################### Aglomerativos
library(ggdendro)

agl_ward <-
  hclust(dados_diss, method = "ward.D2")

plot(cut(as.dendrogram(agl_ward), h = 20)$upper,
     main = "Ward - cortado em H = 20")

agl_ward_res <- cutree(agl_ward, k = 3:8)
#####################################
agl_single <-
  hclust(dados_diss, method = "single")

plot(cut(as.dendrogram(agl_single), h = 4)$upper,
     main = "Vizinho mais Pr ximo - cortado em H = 4",
     xlab = "")

agl_single_res <- cutree(agl_single, k = 3:8)
#####################################
agl_complete <-
  hclust(dados_diss, method = "complete")

```

```
plot(cut(as.dendrogram(agl_complete), h = 10)$upper,
     main = "Vizinho mais Distante - cortado em H = 10")

agl_complete_res <- cutree(agl_complete, k = 3:8)
########################################
agl_ave <-
  hclust(dados_diss, method = "average")

plot(cut(as.dendrogram(agl_ave), h = 15)$upper,
     main = "M dia das Dist ncias - cortado em H = 15")

agl_ave_res <- cutree(agl_ave, k = 3:8)
########################################
agl_cent <-
  hclust(dados_diss, method = "centroid")

plot(cut(as.dendrogram(agl_cent), h = 15)$upper,
     main = "Centr ide - cortado em H = 15")

agl_cent_res <- cutree(agl_cent, k = 3:8)
####################### Divisivo
diana <-
  cluster::diana(
    dados_diss,
    diss = TRUE,
    metric = "euclidean",
    keep.diss = FALSE,
    keep.data = FALSE
  )

plot(cut(as.dendrogram(diana), h = 7)$upper,
     main = "Diana - cortado em H = 7")

diana_res <- cutree(diana, k = 3:8)
```

6.6 BIRCH

```
DataframeProcessing = function(x, thresholdvalue, pagesize){
  x = as.data.frame(x)
  for (i in c(1:nrow(x)))
  {
    MakeaCFTree(as.numeric(x[i, ]), pagesize, thresholdvalue, i)
  }
}

BirchCF = function(x,
                   Type = 'df',
```

```
                    branchingfactor = 4,
                    threshold = 0.15){
  mainlist <<- list()

  pagesize <<- branchingfactor
  if (is.data.frame(x))
  {
    DataframeProcessing(x, threshold, branchingfactor)
    y = mainlist[[1]]

    rm(mainlist, envir = .GlobalEnv)
    rm(pagesize, envir = .GlobalEnv)
    return(y)
  }
}

########################################
#### Function to get the cf tree ######
########################################


MakeaCFTree = function (x, pagesize, threshold, i){
  rowvector = as.numeric(x)#converting the data point into a numeric vector
  if (length(mainlist) == 0)
  {
    #No CF's in the list. it will insert the CF into the node
    n = 1
    rootnode = 0
    childnode = 0
    mainlist[[length(mainlist) + 1]] <<- vector('list', 1)
    mainlist[[length(mainlist)]][[1]] <<-
      list(
        N = n,
        LS = rowvector,
        SS = rowvector * rowvector,
        RN =  rootnode,
        CN =  childnode,
        CNT = 0,
        CI = array(i)
      )
    return(1)

  }
  else
  {
    #If there is already a node in the level 1
    parentnode = 0 #By default parNANAent node is 0 :)(self made :P)
```

```
   calculatenearestnode
    #Write code to calculate ls/n
    # print(calculatenearestnode(parentnode,rowvector,1))
    returnlist = calculatenearestnode(parentnode, rowvector, length(
   mainlist))
    depth = returnlist$depth
    index = returnlist$index
    count = returnlist$count
    Rootnode = returnlist$parentnode
    tempnode = mainlist[[depth]][[index]]
    linearsum = rowvector + tempnode$LS

    squaredsum = (rowvector * rowvector) + tempnode$SS

    n = tempnode$N + 1
    radius = Compute_radius(linearsum, squaredsum, n)#calculate radius
    if (radius > threshold)
    {
      #print('radius is greater than threshold')
      if (count == pagesize)
      {
        splitNode(depth, index)#functiob returns depth and index, based on
   that insert off.:)
        temporarycf = mainlist[[depth]][[index]]
        mainlist[[depth]][[length(mainlist[[depth]]) + 1]] <<-
          list(
            N = 1,
            LS = rowvector,
            SS = (rowvector * rowvector),
            CN = temporarycf$CN,
            CNT = 0,
            RN = temporarycf$RN,
            CI = array(i)
          )
        #print("calling recalculate")
        recalculateCF(depth, index)
      }
      else
      {
        #print('entered2')
        n = 1
        childnode = 0
        len = length(mainlist[[depth]]) + 1
        mainlist[[depth]][[len]] <<- vector('list', 1)

        mainlist[[depth]][[len]] <<-
          list(
```

```
            N = 1,
            LS = rowvector,
            SS = rowvector * rowvector,
            CN = 0,
            CNT = 0,
            RN = Rootnode,
            CI = array(i)
          )
        recalculateCF(depth, len)
      }
      #Condition: 1, check if the branching factor allows you to insert
      #condition: 2, check if you have to make it into two branches
    }
    else
    {
      # print('radius is less than threshold')
      #merge the cf with the data point
      f = mainlist[[depth]][[index]]
      templist = list(
        N = (f$N) + 1,
        LS = f$LS + rowvector,
        SS = f$SS + (rowvector * rowvector),
        CN = f$CN,
        CNT = f$CNT,
        RN = f$RN,
        CI = append(f$CI, i)
      )

      mainlist[[depth]][[index]] <<- templist

      recalculateCF(depth, index)
    }



  }
}

Compute_radius = function (LS, SS, N){
  Y = SS - ((LS * LS) / N)
  return(sqrt(sum(Y) / N))
}

calculatenearestnode = function (parentnode, rowvector, depth){
  min = 0
  index = 0

```

```
149   count = 0
150
151   x = mainlist[[depth]]
152   for (i in c(1:length(x)))
153   {
154     if (length(x[[i]]) != 0)
155     {
156       if (x[[i]]$RN == parentnode)
157       {
158         Centroid = (x[[i]]$LS) / (x[[i]]$N)
159         tempsum = sum((rowvector - Centroid) * (rowvector - Centroid))
160         if (count == 0)
161         {
162           min = tempsum
163
164           index = i
165
166         }
167         if (min > tempsum)
168         {
169           index = i
170           min = tempsum
171
172         }
173         rm(tempsum)
174
175         rm(Centroid)
176
177         count = count + 1
178
179       }
180     }
181   }
182   if (x[[index]]$CNT != 0)
183   {
184     #reccursive function to travel deep into the tree
185     return(calculatenearestnode(index, rowvector, (depth - 1)))
186   }
187   else
188   {
189     return(list(
190       depth = depth,
191       index = index,
192       count = count,
193       parentnode = parentnode
194     ))
195   }
```

```
196
197 }
198
199 splitNode = function(depth, index){
200   if (depth != length(mainlist)) {
201     y = mainlist[[depth]][[index]]$RN
202     mainlist[[depth + 1]][[y]]$CN <<- index
203     index = y
204     depth = depth + 1
205
206
207
208     actualnode = mainlist[[depth]][[index]]
209
210     y = list()
211
212     count = 0
213
214     x = mainlist[[depth]]
215     for (i in c(1:length(x)))
216     {
217       if (x[[i]]$RN == actualnode$RN)
218       {
219         count = count + 1
220       }
221     }
222     if (count == pagesize)
223     {
224       returnlist = splitNode(depth, index)
225       depth = returnlist$depth
226       index = returnlist$index
227       if (mainlist[[depth]][[index]]$CN != 0)
228         return(rearrange(depth, index))
229       else
230         return(list(depth = depth, index = index))
231     }
232     else
233     {
234       return(rearrange(depth, index))
235       #write the code to return back
236
237     }
238   }
239   else
240   {
241     actualnode = mainlist[[depth]][[index]]
242
```

```
243     y = list()
244
245     count = 0
246
247     x = mainlist[[depth]]
248     for (i in c(1:length(x)))
249     {
250       if (x[[i]]$RN == actualnode$RN)
251       {
252         count = count + 1
253       }
254     }
255     if (count == pagesize)
256     {
257       return(createnewnodetop(depth, index))
258       #rearrange(depth,index)
259     }
260   }
261 }
262
263 rearrange = function(depth, index){
264   Mainchild = mainlist[[depth]][[index]]
265
266   y = list()
267
268   distvec = array()
269
270   indexvector = array()
271
272   for (i in c(1:length(mainlist[[depth - 1]])))
273   {
274     if (mainlist[[depth - 1]][[i]]$RN == index)
275     {
276       g = mainlist[[depth - 1]][[i]]$LS
277
278       distvec = append(distvec, sum((as.numeric(g) - as.numeric(
279         rep(0, length(g))
280       )) ^ 2))
281       indexvector = append(indexvector, i)
282     }
283   }
284   distvec = distvec[2:length(distvec)]
285   indexvector = indexvector[2:length(indexvector)]
286   arrayindexes = order(distvec)
287
288   cf1list = arrayindexes[1:ceiling(length(arrayindexes) / 2)]
289   cf2list = arrayindexes[(ceiling(length(arrayindexes) / 2) + 1):(length(
```

```
  arrayindexes))]
CF1index = list()

CF2index = list()

for (i in c(cf1list))
{
  if (length(CF1index) == 0)
  {
    CF1index = mainlist[[depth - 1]][[indexvector[i]]]
    CF1index$CNT = 1
    mainlist[[depth - 1]][[indexvector[i]]]$RN <<- index
  }
  else
  {
    z = mainlist[[depth - 1]][[indexvector[i]]]
    CF1index = list(
      N = (CF1index$N) + z$N,
      LS = CF1index$LS + z$LS,
      SS = CF1index$SS + z$SS,
      CNT = Mainchild$CNT,
      CN = Mainchild$CN,
      RN = Mainchild$RN
    )

    mainlist[[depth - 1]][[indexvector[i]]]$RN <<- index
  }
}

#########

########
for (i in c(cf2list))
{
  if (length(CF2index) == 0)
  {
    CF2index = mainlist[[depth - 1]][[indexvector[i]]]
    mainlist[[depth - 1]][[indexvector[i]]]$RN <<-
      length(mainlist[[depth]]) + 1
  }
  else
  {
    z = mainlist[[depth - 1]][[indexvector[i]]]
    CF2index = list(
      N = (CF2index$N) + z$N,
      LS = CF2index$LS + z$LS,
      SS = CF2index$SS + z$SS,
```

```
336         CNT = Mainchild$CNT,
337         CN = Mainchild$CN,
338         RN = Mainchild$RN
339       )
340
341       mainlist[[depth - 1]][[indexvector[i]]]$RN <<-
342         length(mainlist[[depth]]) + 1
343     }
344   }
345   CF1index$RN = Mainchild$RN
346   mainlist[[depth]][[index]] <<- CF1index
347
348   CF2index$RN = Mainchild$RN
349   mainlist[[depth]][[length(mainlist[[depth]]) + 1]] <<- CF2index
350
351   rm(CF2index, CF1index)
352   return(list(depth = depth - 1, index = Mainchild$CN))
353 }
354
355 createnewnodetop = function(depth, index){
356   depth = length(mainlist)
357
358   distvec = array()
359
360   indexvector = array()
361
362   for (i in c(1:length(mainlist[[depth]])))
363   {
364     if (mainlist[[depth]][[i]]$RN == 0)
365     {
366       g = mainlist[[depth]][[i]]$LS
367
368       distvec = append(distvec, sum((as.numeric(g) - as.numeric(
369         rep(0, length(g))
370       )) ^ 2))
371       indexvector = append(indexvector, i)
372     }
373   }
374   distvec = distvec[2:length(distvec)]
375   indexvector = indexvector[2:length(indexvector)]
376   arrayindexes = order(distvec)
377
378   cf1list = arrayindexes[1:ceiling(length(arrayindexes) / 2)]
379   cf2list = arrayindexes[(ceiling(length(arrayindexes) / 2) + 1):(length(
      arrayindexes))]
380   CF1index = list()
381
```

```
382   CF2index = list()
383
384   for (i in c(cf1list))
385   {
386     if (length(CF1index) == 0)
387     {
388       CF1index = mainlist[[depth]][[indexvector[i]]]
389       CF1index$CNT = 1
390       mainlist[[depth]][[indexvector[i]]]$RN <<- 1
391     }
392     else
393     {
394       z = mainlist[[depth]][[indexvector[i]]]
395       CF1index = list(
396         N = (CF1index$N) + z$N,
397         LS = CF1index$LS + z$LS,
398         SS = CF1index$SS + z$SS,
399         CNT = 1 + CF1index$CNT,
400         CN = 0,
401         RN = 0
402       )
403
404       mainlist[[depth]][[indexvector[i]]]$RN <<- 1
405     }
406   }
407   for (i in c(cf2list))
408   {
409     if (length(CF2index) == 0)
410     {
411       CF2index = mainlist[[depth]][[indexvector[i]]]
412       CF2index$CNT = 1
413       mainlist[[depth]][[indexvector[i]]]$RN <<- 2
414     }
415     else
416     {
417       z = mainlist[[depth]][[indexvector[i]]]
418       CF2index = list(
419         N = (CF2index$N) + z$N,
420         LS = CF2index$LS + z$LS,
421         SS = CF2index$SS + z$SS,
422         CNT = 1 + CF1index$CNT,
423         CN = 0 ,
424         RN = 0
425       )
426
427       mainlist[[depth]][[indexvector[i]]]$RN <<- 2
428     }
```

```
  }
  mainlist[[length(mainlist) + 1]] <<- vector('list', 2)

  mainlist[[depth + 1]][[1]] <<- CF1index

  mainlist[[depth + 1]][[2]] <<- CF2index

  return(list(depth = depth, index = index))
}

recalculateCF = function(depth, index){
  # print('entered recalculate')
  CFindex = list()
  for (i in c(1:length(mainlist[[depth]])))
  {
    if (mainlist[[depth]][[i]]$RN == mainlist[[depth]][[index]]$RN)
    {
      if (length(CFindex) == 0)
      {
        CFindex = mainlist[[depth]][[i]]
        CFindex$CNT = 1
      }
      else
      {
        z = mainlist[[depth]][[i]]
        CFindex = list(
          N = (CFindex$N) + z$N,
          LS = CFindex$LS + z$LS,
          SS = CFindex$SS + z$SS,
          CNT = 1 + CFindex$CNT,
          CN = z$CN,
          RN = 0
        )


      }
    }

  }
  x = mainlist[[depth]][[index]]$RN
  if (depth != length(mainlist)) {
    CFindex$RN = mainlist[[depth + 1]][[x]]$RN
    CFindex$CN = mainlist[[depth + 1]][[x]]$CN
    mainlist[[depth + 1]][[x]] <<- CFindex


    if (mainlist[[depth + 1]][[x]]$RN == 0)
```

```
476     {
477       return(1)
478     }
479     else{
480       recalculateCF(depth + 1, x)
481     }
482   }
483   else
484   {
485     return(1)
486   }
487 }
488 
489 
490 ########################################Clustering Algorithm
    ############################################################
491 
492 Fit = function (Type,
493                 element,
494                 nClusters,
495                 nStart,
496                 iter.max = 100,
497                 method = "complete"){
498   array1 = array()
499 
500   array2 = array()
501 
502   centroid = as.data.frame(matrix(nrow = length(element),
503                                   ncol = length(element[[1]]$LS))) #matrix
    for clustering
504   for (i in c(1:length(element)))
505   {
506     centroid[i, ] = (c(element[[i]]$LS / element[[i]]$N))
507   }
508   if (Type == 'kmeans') {
509     kclust = kmeans(
510       x = centroid,
511       centers = nClusters,
512       nstart = nStart,
513       iter.max = 100
514     )
515     cluster = kclust$cluster
516   }
517   else if (Type == 'hclust') {
518     print("H - Cluster")
519     hcluster = hclust(dist(centroid), method)
520     cluster = cutree(hcluster, nClusters)
```

```
  }
  for (i in c(1:length(element)))
  {
    elementarray = element[[i]]$CI
    for (j in c(1:length(elementarray)))
    {
      array1 = append(array1, elementarray[j])
      array2 = append(array2, cluster[i])
    }
  }
  array1 = array1[2:length(array1)]
  array2 = array2[2:length(array2)]
  x = cbind(array1, array2)
  return(x[order(x[, 1]), ])
}
```

6.7 DBSCAN

```
## selecao do melhor eps
dbscan::kNNdist(dados_diss,
                k = 3, all = FALSE)
## ajuste com o eps selecionado
dbscan <- dbscan::dbscan(dados_norm, eps = 2, minPts = 4)

dados$ruido <-  dbscan$cluster

require(writexl)

write_xlsx(dados, "dados_com-ruido_DBSCAN.xlsx")
```

6.8 Análise dos Ruídos

```
dados_DBSCAN <-
  read_excel("dados_com-ruido_DBSCAN.xlsx")
################################
dados_DBSCAN$out <- ifelse(dados_DBSCAN$ruido == 0, "1", "0")

descritiva_out <-
  round(psych::describe(dados[dados_DBSCAN$out == "1", -c(1:3, 15:16)]), 2)

#################### Gr ficos
outbox_nascidos_vivos <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out, log(
    nascidos_vivos)), fill = "aquamarine4", ) +
  labs(y = "N mero de Nascidos Vivos (log)", x = "Outlier") +
  theme_classic() +
  geom_text(
    data = subset(dados, nascidos_vivos < 5),
```

```
    aes(porc_peso_menor_2500,
        porc_premat,
        label = paste0(municipio, ", ", uf)),
    size = 3,
    check_overlap = TRUE
  )

outbox_porc_premat <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out, porc_
    premat), fill = "aquamarine4", ) +
  labs(y = "Prematuridade (%)", x = "Outlier") +
  theme_classic()

outbox_porc_gesta_multipla <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out,
    porc_gesta_multipla), fill = "aquamarine4") +
  labs(y = "Gesta  o  M ltipla (%)", x = "Outlier") +
  theme_classic()

outbox_porc_0_consulta <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out, porc
    _0_consulta), fill = "aquamarine4") +
  labs(y = "Nenhuma Consulta Pr  -Natal (%)", x = "Outlier") +
  theme_classic()

outbox_porc_7mais_consulta <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out,
    porc_7mais_consulta), fill = "aquamarine4") +
  labs(y = "7 ou mais Consultas Pr  -Natal (%)", x = "Outlier") +
  theme_classic()

outbox_porc_cesarea <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out, porc_
    cesarea), fill = "aquamarine4") +
  labs(y = "Parto Ces rea (%)", x = "Outlier") +
  theme_classic()

outbox_porc_apgar1_menor_7 <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out,
    porc_apgar1_menor_7), fill = "aquamarine4") +
  labs(y = "Apgar do 1   Minuto menor que 7 (%)", x = "Outlier") +
  theme_classic()

outbox_porc_apgar5_menor_7 <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out,
    porc_apgar5_menor_7), fill = "aquamarine4") +
  labs(y = "Apgar do 5   Minuto menor que 7 (%)", x = "Outlier") +
  theme_classic()

outbox_porc_peso_menor_2500 <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out,
     porc_peso_menor_2500), fill = "aquamarine4") +
  labs(y = "Peso menor que 2.500g (%)", x = "Outlier") +
  theme_classic()

```

```
outbox_porc_anomalia <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out, porc_
    anomalia), fill = "aquamarine4") +
  labs(y = "Anomalia Cong nita (%)", x = "Outlier") +
  theme_classic()

outbox_porc_fem <- ggplot(dados_DBSCAN) + geom_boxplot(aes(out, porc_fem),
    fill = "aquamarine4") +
  labs(y = "Sexo Feminino (%)", x = "Outlier") +
  theme_classic()
```

6.9 Agrupamento Espectral

```
spectral_clustering <- function(data, # dados
                                k) # numero de grupos
                                {
  # matriz de dissimilaridade
  A <-
    as.matrix(KRLS::gausskernel(data, 2)) # sigma^2 = 1
  diag(A) <- 0

  # matriz Laplaciana
  L = diag(1 / sqrt(rowSums(A))) %*% A %*% diag(1 / sqrt(rowSums(A)))

  # matriz de autovetores associados aos k maiores autovalores
  auto <- eigen(L, symmetric = TRUE)
  X <- auto$vectors[, 1:k]

  # matriz normalizada
  Y <- X / sqrt(rowSums(X ^ 2))

  # agrupamente k medias na matriz redimensionalizada
  set.seed(1122)
  k_means <- kmeans(Y, centers = k)

  return(list(
    clusters = k_means$cluster, #retornar grupos
    auto = auto,                #retornar autovalores e autovetores
    autovetores = X,            #retornar autovetores dos K maiores
   autovalores
    normalizada = Y             #retornar matriz normalizada
  ))
}
```

6.10 Seleção do Método

```
############################ Medidas de Valida  o
#diana_res[,1]
```

```
diana_res1_index <- data.frame(
  metodo = "diana_res1_index",
  db_index_cent = clusterSim::index.DB(dados_norm, diana_res[, 1])$DB,
  db_index_med = clusterSim::index.DB(dados_norm,
                                      diana_res[, 1],
                                      d = dados_diss,
                                      centrotypes = "medoids")$DB,
  dunn_index = clValid::dunn(distance = dados_diss, diana_res[, 1]),
  silh_index = clusterSim::index.S(dados_diss, diana_res[, 1]),
  ch_index_cent =  clusterSim::index.G1(dados_norm, diana_res[, 1]),
  ch_index_med = clusterSim::index.G1(dados_norm, diana_res[, 1], d = dados
    _diss, centrotypes = "medoids")
)
#diana_res[,2]
diana_res2_index <- data.frame(
  metodo = "diana_res2_index",
  db_index_cent = clusterSim::index.DB(dados_norm, diana_res[, 2])$DB,
  db_index_med = clusterSim::index.DB(dados_norm,
                                      diana_res[, 2],
                                      d = dados_diss,
                                      centrotypes = "medoids")$DB,
  dunn_index = clValid::dunn(distance = dados_diss, diana_res[, 2]),
  silh_index = clusterSim::index.S(dados_diss, diana_res[, 2]),
  ch_index_cent = clusterSim::index.G1(dados_norm, diana_res[, 2]),
  ch_index_med = clusterSim::index.G1(dados_norm, diana_res[, 2], d = dados
    _diss, centrotypes = "medoids")
)
#diana_res[,3]
diana_res3_index <- data.frame(
  metodo = "diana_res3_index",
  db_index_cent = clusterSim::index.DB(dados_norm, diana_res[, 3])$DB,
  db_index_med = clusterSim::index.DB(dados_norm,
                                      diana_res[, 3],
                                      d = dados_diss,
                                      centrotypes = "medoids")$DB,
  dunn_index = clValid::dunn(distance = dados_diss, diana_res[, 3]),
  silh_index = clusterSim::index.S(dados_diss, diana_res[, 3]),
  ch_index_cent = clusterSim::index.G1(dados_norm, diana_res[, 3]),
  ch_index_med = clusterSim::index.G1(dados_norm, diana_res[, 3], d = dados
    _diss, centrotypes = "medoids")
)
#diana_res[,4]
diana_res4_index <- data.frame(
  metodo = "diana_res4_index",
  db_index_cent = clusterSim::index.DB(dados_norm, diana_res[, 4])$DB,
  db_index_med = clusterSim::index.DB(dados_norm,
                                      diana_res[, 4],
```

```
                                        d = dados_diss,
                                        centrotypes = "medoids")$DB,
  dunn_index = clValid::dunn(distance = dados_diss, diana_res[, 4]),
  silh_index = clusterSim::index.S(dados_diss, diana_res[, 4]),
  ch_index_cent = clusterSim::index.G1(dados_norm, diana_res[, 4]),
  ch_index_med = clusterSim::index.G1(dados_norm, diana_res[, 4], d = dados
    _diss, centrotypes = "medoids")
)
#diana_res[,5]
diana_res5_index <- data.frame(
  metodo = "diana_res5_index",
  db_index_cent = clusterSim::index.DB(dados_norm, diana_res[, 5])$DB,
  db_index_med = clusterSim::index.DB(dados_norm,
                                        diana_res[, 5],
                                        d = dados_diss,
                                        centrotypes = "medoids")$DB,
  dunn_index = clValid::dunn(distance = dados_diss, diana_res[, 5]),
  silh_index = clusterSim::index.S(dados_diss, diana_res[, 5]),
  ch_index_cent = clusterSim::index.G1(dados_norm, diana_res[, 5]),
  ch_index_med = clusterSim::index.G1(dados_norm, diana_res[, 5], d = dados
    _diss, centrotypes = "medoids")
)
#diana_res[,6]
diana_res6_index <- data.frame(
  metodo = "diana_res6_index",
  db_index_cent = clusterSim::index.DB(dados_norm, diana_res[, 6])$DB,
  db_index_med = clusterSim::index.DB(dados_norm,
                                        diana_res[, 6],
                                        d = dados_diss,
                                        centrotypes = "medoids")$DB,
  dunn_index = clValid::dunn(distance = dados_diss, diana_res[, 6]),
  silh_index = clusterSim::index.S(dados_diss, diana_res[, 6]),
  ch_index_cent = clusterSim::index.G1(dados_norm, diana_res[, 6]),
  ch_index_med = clusterSim::index.G1(dados_norm, diana_res[, 6], d = dados
    _diss, centrotypes = "medoids")

)
#agl_cent_res[,1]
agl_cent_res1_index <- data.frame(
  metodo = "agl_cent_res1_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_cent_res[, 1])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_cent_res[, 1],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
```

```
  dunn_index = clValid::dunn(distance = dados_diss, agl_cent_res[, 1]),
  silh_index = clusterSim::index.S(dados_diss, agl_cent_res[, 1]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_cent_res[, 1]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_cent_res[, 1], d =
    dados_diss, centrotypes = "medoids")
)
#agl_cent_res[,2]
agl_cent_res2_index <- data.frame(
  metodo = "agl_cent_res2_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_cent_res[, 2])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_cent_res[, 2],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_cent_res[, 2]),
  silh_index = clusterSim::index.S(dados_diss, agl_cent_res[, 2]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_cent_res[, 2]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_cent_res[, 2], d =
    dados_diss, centrotypes = "medoids")
)
#agl_cent_res[,3]
agl_cent_res3_index <- data.frame(
  metodo = "agl_cent_res3_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_cent_res[, 3])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_cent_res[, 3],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_cent_res[, 3]),
  silh_index = clusterSim::index.S(dados_diss, agl_cent_res[, 3]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_cent_res[, 3]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_cent_res[, 3], d =
    dados_diss, centrotypes = "medoids")
)
#agl_cent_res[,4]
agl_cent_res4_index <- data.frame(
  metodo = "agl_cent_res4_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_cent_res[, 4])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_cent_res[, 4],
    d = dados_diss,
    centrotypes = "medoids"
```

```
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_cent_res[, 4]),
  silh_index = clusterSim::index.S(dados_diss, agl_cent_res[, 4]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_cent_res[, 4]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_cent_res[, 4], d =
    dados_diss, centrotypes = "medoids")
)
#agl_cent_res[,5]
agl_cent_res5_index <- data.frame(
  metodo = "agl_cent_res5_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_cent_res[, 5])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_cent_res[, 5],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_cent_res[, 5]),
  silh_index = clusterSim::index.S(dados_diss, agl_cent_res[, 5]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_cent_res[, 5]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_cent_res[, 5], d =
    dados_diss, centrotypes = "medoids")
)
#agl_cent_res[,6]
agl_cent_res6_index <- data.frame(
  metodo = "agl_cent_res6_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_cent_res[, 6])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_cent_res[, 6],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_cent_res[, 6]),
  silh_index = clusterSim::index.S(dados_diss, agl_cent_res[, 6]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_cent_res[, 6]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_cent_res[, 6], d =
    dados_diss, centrotypes = "medoids")
)
#agl_ward_res[,1]
agl_ward_res1_index <- data.frame(
  metodo = "agl_ward_res1_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_ward_res[, 1])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_ward_res[, 1],
    d = dados_diss,
```

```
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_ward_res[, 1]),
  silh_index = clusterSim::index.S(dados_diss, agl_ward_res[, 1]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_ward_res[, 1]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_ward_res[, 1], d =
    dados_diss, centrotypes = "medoids")
)
#agl_ward_res[,2]
agl_ward_res2_index <- data.frame(
  metodo = "agl_ward_res2_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_ward_res[, 2])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_ward_res[, 2],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_ward_res[, 2]),
  silh_index = clusterSim::index.S(dados_diss, agl_ward_res[, 2]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_ward_res[, 2]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_ward_res[, 2], d =
    dados_diss, centrotypes = "medoids")
)
#agl_ward_res[,3]
agl_ward_res3_index <- data.frame(
  metodo = "agl_ward_res3_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_ward_res[, 3])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_ward_res[, 3],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_ward_res[, 3]),
  silh_index = clusterSim::index.S(dados_diss, agl_ward_res[, 3]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_ward_res[, 3]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_ward_res[, 3], d =
    dados_diss, centrotypes = "medoids")
)
#agl_ward_res[,4]
agl_ward_res4_index <- data.frame(
  metodo = "agl_ward_res4_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_ward_res[, 4])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_ward_res[, 4],
```

```
223     d = dados_diss,
224     centrotypes = "medoids"
225   )$DB,
226   dunn_index = clValid::dunn(distance = dados_diss, agl_ward_res[, 4]),
227   silh_index = clusterSim::index.S(dados_diss, agl_ward_res[, 4]),
228   ch_index_cent = clusterSim::index.G1(dados_norm, agl_ward_res[, 4]),
229   ch_index_med = clusterSim::index.G1(dados_norm, agl_ward_res[, 4], d =
      dados_diss, centrotypes = "medoids")
230
231 )
232 #agl_ward_res[,5]
233 agl_ward_res5_index <- data.frame(
234   metodo = "agl_ward_res5_index",
235   db_index_cent = clusterSim::index.DB(dados_norm, agl_ward_res[, 5])$DB,
236   db_index_med = clusterSim::index.DB(
237     dados_norm,
238     agl_ward_res[, 5],
239     d = dados_diss,
240     centrotypes = "medoids"
241   )$DB,
242   dunn_index = clValid::dunn(distance = dados_diss, agl_ward_res[, 5]),
243   silh_index = clusterSim::index.S(dados_diss, agl_ward_res[, 5]),
244   ch_index_cent = clusterSim::index.G1(dados_norm, agl_ward_res[, 5]),
245   ch_index_med = clusterSim::index.G1(dados_norm, agl_ward_res[, 5], d =
      dados_diss, centrotypes = "medoids")
246 )
247 #agl_ward_res[,6]
248 agl_ward_res6_index <- data.frame(
249   metodo = "agl_ward_res6_index",
250   db_index_cent = clusterSim::index.DB(dados_norm, agl_ward_res[, 6])$DB,
251   db_index_med = clusterSim::index.DB(
252     dados_norm,
253     agl_ward_res[, 6],
254     d = dados_diss,
255     centrotypes = "medoids"
256   )$DB,
257   dunn_index = clValid::dunn(distance = dados_diss, agl_ward_res[, 6]),
258   silh_index = clusterSim::index.S(dados_diss, agl_ward_res[, 6]),
259   ch_index_cent = clusterSim::index.G1(dados_norm, agl_ward_res[, 6]),
260   ch_index_med = clusterSim::index.G1(dados_norm, agl_ward_res[, 6], d =
      dados_diss, centrotypes = "medoids")
261 )
262 #agl_complete_res[,1]
263 agl_complete_res1_index <-
264   data.frame(
265     metodo = "agl_complete_res1_index",
266     db_index_cent = clusterSim::index.DB(dados_norm, agl_complete_res[, 1])
```

```
    $DB,
267     db_index_med = clusterSim::index.DB(
268       dados_norm,
269       agl_complete_res[, 1],
270       d = dados_diss,
271       centrotypes = "medoids"
272     )$DB,
273     dunn_index = clValid::dunn(distance = dados_diss, agl_complete_res[,
    1]),
274     silh_index = clusterSim::index.S(dados_diss, agl_complete_res[, 1]),
275     ch_index_cent = clusterSim::index.G1(dados_norm, agl_complete_res[, 1])
    ,
276     ch_index_med = clusterSim::index.G1(dados_norm, agl_complete_res[, 1],
    d = dados_diss, centrotypes = "medoids")
277   )
278 #agl_complete_res[,2]
279 agl_complete_res2_index <-
280   data.frame(
281     metodo = "agl_complete_res2_index",
282     db_index_cent = clusterSim::index.DB(dados_norm, agl_complete_res[, 2])
    $DB,
283     db_index_med = clusterSim::index.DB(
284       dados_norm,
285       agl_complete_res[, 2],
286       d = dados_diss,
287       centrotypes = "medoids"
288     )$DB,
289     dunn_index = clValid::dunn(distance = dados_diss, agl_complete_res[,
    2]),
290     silh_index = clusterSim::index.S(dados_diss, agl_complete_res[, 2]),
291     ch_index_cent = clusterSim::index.G1(dados_norm, agl_complete_res[, 2])
    ,
292     ch_index_med = clusterSim::index.G1(dados_norm, agl_complete_res[, 2],
    d = dados_diss, centrotypes = "medoids")
293
294   )
295 #agl_complete_res[,3]
296 agl_complete_res3_index <-
297   data.frame(
298     metodo = "agl_complete_res3_index",
299     db_index_cent = clusterSim::index.DB(dados_norm, agl_complete_res[, 3])
    $DB,
300     db_index_med = clusterSim::index.DB(
301       dados_norm,
302       agl_complete_res[, 3],
303       d = dados_diss,
304       centrotypes = "medoids"
```

```
305     )$DB,
306     dunn_index = clValid::dunn(distance = dados_diss, agl_complete_res[,
    3]),
307     silh_index = clusterSim::index.S(dados_diss, agl_complete_res[, 3]),
308     ch_index_cent = clusterSim::index.G1(dados_norm, agl_complete_res[, 3])
    ,
309     ch_index_med = clusterSim::index.G1(dados_norm, agl_complete_res[, 3],
    d = dados_diss, centrotypes = "medoids")
310   )
311 #agl_complete_res[,4]
312 agl_complete_res4_index <-
313   data.frame(
314     metodo = "agl_complete_res4_index",
315     db_index_cent = clusterSim::index.DB(dados_norm, agl_complete_res[, 4])
    $DB,
316     db_index_med = clusterSim::index.DB(
317       dados_norm,
318       agl_complete_res[, 4],
319       d = dados_diss,
320       centrotypes = "medoids"
321     )$DB,
322     dunn_index = clValid::dunn(distance = dados_diss, agl_complete_res[,
    4]),
323     silh_index = clusterSim::index.S(dados_diss, agl_complete_res[, 4]),
324     ch_index_cent = clusterSim::index.G1(dados_norm, agl_complete_res[, 4])
    ,
325     ch_index_med = clusterSim::index.G1(dados_norm, agl_complete_res[, 4],
    d = dados_diss, centrotypes = "medoids")
326   )
327 #agl_complete_res[,5]
328 agl_complete_res5_index <-
329   data.frame(
330     metodo = "agl_complete_res5_index",
331     db_index_cent = clusterSim::index.DB(dados_norm, agl_complete_res[, 5])
    $DB,
332     db_index_med = clusterSim::index.DB(
333       dados_norm,
334       agl_complete_res[, 5],
335       d = dados_diss,
336       centrotypes = "medoids"
337     )$DB,
338     dunn_index = clValid::dunn(distance = dados_diss, agl_complete_res[,
    5]),
339     silh_index = clusterSim::index.S(dados_diss, agl_complete_res[, 5]),
340     ch_index_cent = clusterSim::index.G1(dados_norm, agl_complete_res[, 5])
    ,
341     ch_index_med = clusterSim::index.G1(dados_norm, agl_complete_res[, 5],
```

```
    d = dados_diss, centrotypes = "medoids")
  )
#agl_complete_res[,6]
agl_complete_res6_index <-
  data.frame(
    metodo = "agl_complete_res6_index",
    db_index_cent = clusterSim::index.DB(dados_norm, agl_complete_res[, 6])
    $DB,
    db_index_med = clusterSim::index.DB(
      dados_norm,
      agl_complete_res[, 6],
      d = dados_diss,
      centrotypes = "medoids"
    )$DB,
    dunn_index = clValid::dunn(distance = dados_diss, agl_complete_res[,
    6]),
    silh_index = clusterSim::index.S(dados_diss, agl_complete_res[, 6]),
    ch_index_cent = clusterSim::index.G1(dados_norm, agl_complete_res[, 6])
    ,
    ch_index_med = clusterSim::index.G1(dados_norm, agl_complete_res[, 6],
    d = dados_diss, centrotypes = "medoids")
  )
#agl_ave_res[,1]
agl_ave_res1_index <- data.frame(
  metodo = "agl_ave_res1_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_ave_res[, 1])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_ave_res[, 1],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_ave_res[, 1]),
  silh_index = clusterSim::index.S(dados_diss, agl_ave_res[, 1]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_ave_res[, 1]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_ave_res[, 1], d =
    dados_diss, centrotypes = "medoids")
)
#agl_ave_res[,2]
agl_ave_res2_index <- data.frame(
  metodo = "agl_ave_res2_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_ave_res[, 2])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_ave_res[, 2],
    d = dados_diss,
    centrotypes = "medoids"
```

```
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_ave_res[, 2]),
  silh_index = clusterSim::index.S(dados_diss, agl_ave_res[, 2]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_ave_res[, 2]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_ave_res[, 2], d =
    dados_diss, centrotypes = "medoids")
)
#agl_ave_res[,3]
agl_ave_res3_index <- data.frame(
  metodo = "agl_ave_res3_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_ave_res[, 3])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_ave_res[, 3],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_ave_res[, 3]),
  silh_index = clusterSim::index.S(dados_diss, agl_ave_res[, 3]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_ave_res[, 3]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_ave_res[, 3], d =
    dados_diss, centrotypes = "medoids")
)
#agl_ave_res[,4]
agl_ave_res4_index <- data.frame(
  metodo = "agl_ave_res4_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_ave_res[, 4])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_ave_res[, 4],
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, agl_ave_res[, 4]),
  silh_index = clusterSim::index.S(dados_diss, agl_ave_res[, 4]),
  ch_index_cent = clusterSim::index.G1(dados_norm, agl_ave_res[, 4]),
  ch_index_med = clusterSim::index.G1(dados_norm, agl_ave_res[, 4], d =
    dados_diss, centrotypes = "medoids")
)
#agl_ave_res[,5]
agl_ave_res5_index <- data.frame(
  metodo = "agl_ave_res5_index",
  db_index_cent = clusterSim::index.DB(dados_norm, agl_ave_res[, 5])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    agl_ave_res[, 5],
    d = dados_diss,
```

```
427     centrotypes = "medoids"
428   )$DB,
429   dunn_index = clValid::dunn(distance = dados_diss, agl_ave_res[, 5]),
430   silh_index = clusterSim::index.S(dados_diss, agl_ave_res[, 5]),
431   ch_index_cent = clusterSim::index.G1(dados_norm, agl_ave_res[, 5]),
432   ch_index_med = clusterSim::index.G1(dados_norm, agl_ave_res[, 5], d =
      dados_diss, centrotypes = "medoids")
433 )
434 #agl_ave_res[,6]
435 agl_ave_res6_index <- data.frame(
436   metodo = "agl_ave_res6_index",
437   db_index_cent = clusterSim::index.DB(dados_norm, agl_ave_res[, 6])$DB,
438   db_index_med = clusterSim::index.DB(
439     dados_norm,
440     agl_ave_res[, 6],
441     d = dados_diss,
442     centrotypes = "medoids"
443   )$DB,
444   dunn_index = clValid::dunn(distance = dados_diss, agl_ave_res[, 6]),
445   silh_index = clusterSim::index.S(dados_diss, agl_ave_res[, 6]),
446   ch_index_cent = clusterSim::index.G1(dados_norm, agl_ave_res[, 6]),
447   ch_index_med = clusterSim::index.G1(dados_norm, agl_ave_res[, 6], d =
      dados_diss, centrotypes = "medoids")
448
449 )
450 #agl_single_res[,1]
451 agl_single_res1_index <-
452   data.frame(
453     metodo = "agl_single_res1_index",
454     db_index_cent = clusterSim::index.DB(dados_norm, agl_single_res[, 1])$
      DB,
455     db_index_med = clusterSim::index.DB(
456       dados_norm,
457       agl_single_res[, 1],
458       d = dados_diss,
459       centrotypes = "medoids"
460     )$DB,
461     dunn_index = clValid::dunn(distance = dados_diss, agl_single_res[, 1]),
462     silh_index = clusterSim::index.S(dados_diss, agl_single_res[, 1]),
463     ch_index_cent = clusterSim::index.G1(dados_norm, agl_single_res[, 1]),
464     ch_index_med = clusterSim::index.G1(dados_norm, agl_single_res[, 1], d
      = dados_diss, centrotypes = "medoids")
465   )
466 #agl_single_res[,2]
467 agl_single_res2_index <-
468   data.frame(
469     metodo = "agl_single_res2_index",
```

```
470     db_index_cent = clusterSim::index.DB(dados_norm, agl_single_res[, 2])$
      DB,
471     db_index_med = clusterSim::index.DB(
472       dados_norm,
473       agl_single_res[, 2],
474       d = dados_diss,
475       centrotypes = "medoids"
476     )$DB,
477     dunn_index = clValid::dunn(distance = dados_diss, agl_single_res[, 2]),
478     silh_index = clusterSim::index.S(dados_diss, agl_single_res[, 2]),
479     ch_index_cent = clusterSim::index.G1(dados_norm, agl_single_res[, 2]),
480     ch_index_med = clusterSim::index.G1(dados_norm, agl_single_res[, 2], d
      = dados_diss, centrotypes = "medoids")
481   )
482 #agl_single_res[,3]
483 agl_single_res3_index <-
484   data.frame(
485     metodo = "agl_single_res3_index",
486     db_index_cent = clusterSim::index.DB(dados_norm, agl_single_res[, 3])$
      DB,
487     db_index_med = clusterSim::index.DB(
488       dados_norm,
489       agl_single_res[, 3],
490       d = dados_diss,
491       centrotypes = "medoids"
492     )$DB,
493     dunn_index = clValid::dunn(distance = dados_diss, agl_single_res[, 3]),
494     silh_index = clusterSim::index.S(dados_diss, agl_single_res[, 3]),
495     ch_index_cent = clusterSim::index.G1(dados_norm, agl_single_res[, 3]),
496     ch_index_med = clusterSim::index.G1(dados_norm, agl_single_res[, 3], d
      = dados_diss, centrotypes = "medoids")
497   )
498 #agl_single_res[,4]
499 agl_single_res4_index <-
500   data.frame(
501     metodo = "agl_single_res4_index",
502     db_index_cent = clusterSim::index.DB(dados_norm, agl_single_res[, 4])$
      DB,
503     db_index_med = clusterSim::index.DB(
504       dados_norm,
505       agl_single_res[, 4],
506       d = dados_diss,
507       centrotypes = "medoids"
508     )$DB,
509     dunn_index = clValid::dunn(distance = dados_diss, agl_single_res[, 4]),
510     silh_index = clusterSim::index.S(dados_diss, agl_single_res[, 4]),
511     ch_index_cent = clusterSim::index.G1(dados_norm, agl_single_res[, 4]),
```

```
512     ch_index_med = clusterSim::index.G1(dados_norm, agl_single_res[, 4], d
      = dados_diss, centrotypes = "medoids")
513   )
514 #agl_single_res[,5]
515 agl_single_res5_index <-
516   data.frame(
517     metodo = "agl_single_res5_index",
518     db_index_cent = clusterSim::index.DB(dados_norm, agl_single_res[, 5])$
      DB,
519     db_index_med = clusterSim::index.DB(
520       dados_norm,
521       agl_single_res[, 5],
522       d = dados_diss,
523       centrotypes = "medoids"
524     )$DB,
525     dunn_index = clValid::dunn(distance = dados_diss, agl_single_res[, 5]),
526     silh_index = clusterSim::index.S(dados_diss, agl_single_res[, 5]),
527     ch_index_cent = clusterSim::index.G1(dados_norm, agl_single_res[, 5]),
528     ch_index_med = clusterSim::index.G1(dados_norm, agl_single_res[, 5], d
      = dados_diss, centrotypes = "medoids")
529   )
530 #agl_single_res[,6]
531 agl_single_res6_index <-
532   data.frame(
533     metodo = "agl_single_res6_index",
534     db_index_cent = clusterSim::index.DB(dados_norm, agl_single_res[, 6])$
      DB,
535     db_index_med = clusterSim::index.DB(
536       dados_norm,
537       agl_single_res[, 6],
538       d = dados_diss,
539       centrotypes = "medoids"
540     )$DB,
541     dunn_index = clValid::dunn(distance = dados_diss, agl_single_res[, 6]),
542     silh_index = clusterSim::index.S(dados_diss, agl_single_res[, 6]),
543     ch_index_cent = clusterSim::index.G1(dados_norm, agl_single_res[, 6]),
544     ch_index_med = clusterSim::index.G1(dados_norm, agl_single_res[, 6], d
      = dados_diss, centrotypes = "medoids")
545   )
546 #birch_k_medias$array2
547 birch_k_medias_index <- data.frame(
548   metodo = "birch_k_medias",
549   db_index_cent = clusterSim::index.DB(dados_norm, birch_k_medias$array2)$
      DB,
550   db_index_med = clusterSim::index.DB(
551     dados_norm,
552     birch_k_medias$array2,
```

```
553     d = dados_diss,
554     centrotypes = "medoids"
555   )$DB,
556   dunn_index = clValid::dunn(distance = dados_diss, birch_k_medias$array2),
557   silh_index = clusterSim::index.S(dados_diss, birch_k_medias$array2),
558   ch_index_cent = clusterSim::index.G1(dados_norm, birch_k_medias$array2),
559   ch_index_med = clusterSim::index.G1(dados_norm, birch_k_medias$array2, d
      = dados_diss, centrotypes = "medoids")
560 )
561 #espectral$clusters
562 espectral_index <- data.frame(
563   metodo = "espectral",
564   db_index_cent = clusterSim::index.DB(dados_norm, espectral$clusters)$DB,
565   db_index_med = clusterSim::index.DB(
566     dados_norm,
567     espectral$clusters,
568     d = dados_diss,
569     centrotypes = "medoids"
570   )$DB,
571   dunn_index = clValid::dunn(distance = dados_diss, espectral$clusters),
572   silh_index = clusterSim::index.S(dados_diss, espectral$clusters),
573   ch_index_cent = clusterSim::index.G1(dados_norm, espectral$clusters),
574   ch_index_med = clusterSim::index.G1(dados_norm, espectral$clusters, d =
      dados_diss, centrotypes = "medoids")
575 )
576 #pam$clustering
577 pam_index <- data.frame(
578   metodo = "pam",
579   db_index_cent = clusterSim::index.DB(dados_norm, pam$clustering)$DB,
580   db_index_med = clusterSim::index.DB(
581     dados_norm,
582     pam$clustering,
583     d = dados_diss,
584     centrotypes = "medoids"
585   )$DB,
586   dunn_index = clValid::dunn(distance = dados_diss, pam$clustering),
587   silh_index = clusterSim::index.S(dados_diss, pam$clustering),
588   ch_index_cent = clusterSim::index.G1(dados_norm, pam$clustering),
589   ch_index_med = clusterSim::index.G1(dados_norm, pam$clustering, d = dados
      _diss, centrotypes = "medoids")
590 )
591 #clara$clustering
592 clara_7_index <- data.frame(
593   metodo = "clara",
594   db_index_cent = clusterSim::index.DB(dados_norm, clara$clustering)$DB,
595   db_index_med = clusterSim::index.DB(
596     dados_norm,
```

```
    clara$clustering,
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, clara$clustering),
  silh_index = clusterSim::index.S(dados_diss, clara$clustering),
  ch_index_cent = clusterSim::index.G1(dados_norm, clara$clustering),
  ch_index_med = clusterSim::index.G1(dados_norm, clara$clustering, d =
    dados_diss, centrotypes = "medoids")
)
#k_medias$cluster
k_medias_index <- data.frame(
  metodo = "k_medias",
  db_index_cent = clusterSim::index.DB(dados_norm, k_medias$cluster)$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm,
    k_medias$cluster,
    d = dados_diss,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss, k_medias$cluster),
  silh_index = clusterSim::index.S(dados_diss, k_medias$cluster),
  ch_index_cent = clusterSim::index.G1(dados_norm, k_medias$cluster),
  ch_index_med = clusterSim::index.G1(dados_norm, k_medias$cluster, d =
    dados_diss, centrotypes = "medoids")
)

############################## dbscan
dados_diss_dbscan <- dist(dados_norm[dados_DBSCAN$out == "0",],
                          method = "euclidean")
dbscan_index <- data.frame(
  metodo = "dbscan",
  db_index_cent = clusterSim::index.DB(dados_norm[dados_DBSCAN$out == "0"
    ,],
                                       dados_DBSCAN$ruido[dados_DBSCAN$out
    == "0"])$DB,
  db_index_med = clusterSim::index.DB(
    dados_norm[dados_DBSCAN$out == "0",],
    dados_DBSCAN$ruido[dados_DBSCAN$out == "0"],
    d = dados_diss_dbscan,
    centrotypes = "medoids"
  )$DB,
  dunn_index = clValid::dunn(distance = dados_diss_dbscan,
                             dados_DBSCAN$ruido[dados_DBSCAN$out == "0"]),
  silh_index = clusterSim::index.S(dados_diss_dbscan,
                                   dados_DBSCAN$ruido[dados_DBSCAN$out == "
    0"]),
```

```
  ch_index_cent = clusterSim::index.G1(dados_norm[dados_DBSCAN$out == "0",],
                                       dados_DBSCAN$ruido[dados_DBSCAN$out == "0"]),
  ch_index_med = clusterSim::index.G1(dados_norm[dados_DBSCAN$out == "0",],
                                      dados_DBSCAN$ruido[dados_DBSCAN$out == "0"],
                                      d = dados_diss_dbscan, centrotypes = "medoids")
)
```

6.11 Análise do Resultado

```
########################## Analise Resultado

dados$cluster <- k_medias$cluster

media <- function(x)
  mean(x, na.rm = TRUE)
medi <- function(x)
  median(x, na.rm = TRUE)
dp <- function(x)
  sd(x, na.rm = TRUE)
mini <- function(x)
  min(x, na.rm = TRUE)
maxi <- function(x)
  max(x, na.rm = TRUE)
n <- function(x)
  sum(!is.na(x))

datasummary(
  as.factor(cluster) ~ (
    nascidos_vivos + porc_premat + porc_gesta_multipla + porc_cesarea +
      porc_0_consulta + porc_7mais_consulta + porc_apgar1_menor_7 +
      porc_apgar5_menor_7 + porc_anomalia + porc_peso_menor_2500 + porc_fem
  ) * (n + media + dp + mini + medi + maxi),
  data = dados
)

################  rvore  de classifica  o

dados_arv <- dados %>%
  rename(
    premat = porc_premat,
    multipla = porc_gesta_multipla,
    cesarea = porc_cesarea,
    cons0 = porc_0_consulta,
```

```
    cons7 = porc_7mais_consulta,
    ap1 =  porc_apgar1_menor_7,
    ap5 = porc_apgar5_menor_7,
    anomalia = porc_anomalia,
    peso = porc_peso_menor_2500,
    fem = porc_fem
  )

arvore <- rpart(as.factor(cluster) ~ ., data = dados_arv[, -c(1:4)])
prp(arvore)

################ constru  o mapa
mapa_grups <-
  left_join(mapa_dados_df, dados[, c("codigo", "cluster")])

memory.limit(size = 100000)
mapa_cluster <-
  ggplot(mapa_grups %>%
           filter(codigo %in% dados$codigo)) +
  theme_minimal() +
  geom_polygon(aes(
    x = long,
    y = lat,
    group = group,
    fill = as.factor(cluster)
  )) +
  coord_equal() +
  theme_minimal() +
  scale_fill_viridis_d("") +
  theme(legend.position = 'bottom') +
  ggtitle("Grupos") +
  geom_path(
    aes(long, lat, group = group),
    color = "black",
    size = 0.05,
    data = mapa_uf_df
  )

box_nasc <- ggplot(dados[dados$codigo %in% socioeconomicos$codmun6, ]) +
  geom_boxplot(aes(as.factor(cluster), log(nascidos_vivos)), fill = "
    aquamarine4") +
  labs(y = "", x = "Grupo", title = "Nascidos Vivos (log)") + theme_classic
    ()

################## an lise socioecon mica

library(readxl)
```

```r
socioeconomicos <- read_excel("indic_censo_2010_muni.xlsx")

socioeconomicos <- socioeconomicos %>%
  clean_names %>%
  mutate(codmun6 = as.factor(codmun6)) %>%
  select("ano",
         "uf",
         "codmun6",
         "municipio",
         "fectot",
         "gini",
         "rdpc",
         "idhm")

socioeconomicos <-
  left_join(socioeconomicos, dados[, c("codigo", "cluster")],
            by = c("codmun6" = "codigo"))


### Gr ficos de Box-plot

box_fectot <- ggplot(socioeconomicos) +
  geom_boxplot(aes(as.factor(cluster), fectot), fill = "aquamarine4") +
  labs(y = "", x = "Grupo", title = "Fecundidade Total") + theme_classic()

box_gini <- ggplot(socioeconomicos) +
  geom_boxplot(aes(as.factor(cluster), gini), fill = "aquamarine4") +
  labs(y = "", x = "Grupo", title = " ndice  de Gini") + theme_classic()

box_rdpc <- ggplot(socioeconomicos) +
  geom_boxplot(aes(as.factor(cluster), rdpc), fill = "aquamarine4") +
  labs(y = "", x = "Grupo", title = "Renda per Capita M dia") + theme_
    classic()

box_idhm <- ggplot(socioeconomicos) +
  geom_boxplot(aes(as.factor(cluster), idhm), fill = "aquamarine4") +
  labs(y = "", x = "Grupo", title = " ndice  de Desenvolvimento Humano
    Municipal") + theme_classic()
```

6.12 Estudo de Simulação

6.12.1 Funções

```r
# Fun  es para serem utilizadas no estudo de simula  o
cria_dados <- function(semente,
                       cenario,
                       n_k = 50,
```

```
                        k = 7, phi = 150,
                        mu = c(0.01, 0.02, 0.05, 0.10, 0.4, 0.6, 0.75),
                        betas = c(1, 1, 1, 1, 1, -1, -1),
                        ruido = FALSE,
                        var_ruido = 3){

  intercepto <- log(mu/(1 - mu)) %>%
    rep(each = n_k * k) %>%
    matrix(ncol = k)

  if (cenario == 1){
    preditor_linear <- kronecker(diag(betas), rep(1, n_k))
  }
  else {
    diag_betas <- diag(betas)
    diag_betas[1, 2] <- diag_betas[2, 3] <- 1
    diag_betas[3, 4] <- diag_betas[4, 5] <- 1
    diag_betas[5, 6] <- diag_betas[6, 7] <- -1

    preditor_linear <- kronecker(diag_betas, rep(1, n_k))
  }

  medias <- exp(intercepto + preditor_linear) /
    (1 + exp(intercepto + preditor_linear))

  set.seed(semente)
  dados <- apply(medias, 2, function(a){
    rbeta(n_k * k,
          a * phi,
          (1 - a) * phi)
  })
  if (ruido){
    vars_ruido <- sapply(1:var_ruido, function(aa){
      media_aleatoria <- runif(1)
      rbeta(n_k * k,
            media_aleatoria * phi,
            (1 - media_aleatoria) * phi)
    })
    dados <- cbind(dados, vars_ruido)
  }
  dados
}

cria_agrupamentos <- function(dados_agrupar,
                              n_clusters,
                              dados_norm = scale(dados_agrupar),
                              dados_dist = dist(dados_agrupar),
```

```
52                                samples_clara = 10,
53                                nStart_birch = 10,
54                                threshold_birch = 0.15,
55                                eps_dbscan = 1.5,
56                                minPts_dbscan = 5){
57
58  ini_k <- Sys.time()
59  kmeans <-
60    kmeans(dados_norm,
61           centers = n_clusters)$cluster
62  fim_k <- Sys.time()
63  temp_kmeans <- fim_k - ini_k
64
65  ini_pam <- Sys.time()
66  pam <- cluster::pam(dados_norm,
67                      k = n_clusters)$clustering
68  fim_pam <- Sys.time()
69  temp_pam <- fim_pam - ini_pam
70
71  ini_clara <- Sys.time()
72  clara <- cluster::clara(dados_norm,
73                          k = n_clusters,
74                          samples = samples_clara)$clustering
75  fim_clara <- Sys.time()
76  temp_clara <- fim_clara - ini_clara
77
78  ini_ward <- Sys.time()
79  agl_ward <- dados_dist %>%
80     hclust(method = "ward.D") %>%
81     cutree(k = n_clusters)
82  fim_ward <- Sys.time()
83  temp_ward <- fim_ward - ini_ward
84
85  ini_single <- Sys.time()
86  agl_single <-
87    dados_dist %>%
88    hclust(method = "single") %>%
89    cutree(k = n_clusters)
90  fim_single <- Sys.time()
91  temp_single <- fim_single - ini_single
92
93
94  ini_complete <- Sys.time()
95  agl_complete <-
96    dados_dist %>%
97    hclust(method = "complete") %>%
98    cutree(k = n_clusters)
```

```
fim_complete <- Sys.time()
temp_complete <- fim_complete - ini_complete


ini_average <- Sys.time()
agl_average <-
  dados_dist %>%
  hclust(method = "average") %>%
  cutree(k = n_clusters)
fim_average <- Sys.time()
temp_average <- fim_average - ini_average

ini_centroid <- Sys.time()
agl_centroid <-
  dados_dist %>%
  hclust(method = "centroid") %>%
  cutree(k = n_clusters)
fim_centroid <- Sys.time()
temp_centroid <- fim_centroid - ini_centroid

ini_diana <- Sys.time()
diana <-
  dados_dist %>%
  cluster::diana(
    diss = TRUE,
    metric = "euclidean",
    keep.diss = FALSE,
    keep.data = FALSE
  ) %>% cutree(diana, k = n_clusters)
fim_diana <- Sys.time()
temp_diana <- fim_diana - ini_diana


ini_birch <- Sys.time()
CFTree_birch <- as.data.frame(dados_norm) %>%
  BirchCF(threshold = threshold_birch)

birch_k_medias <-  as.data.frame(
  Fit('kmeans', CFTree_birch,
      nClusters = n_clusters,
      nStart = nStart_birch))$array2
fim_birch <- Sys.time()
temp_birch <- fim_birch - ini_birch

ini_dbscan <- Sys.time()
dbscan <- dbscan::dbscan(dados_norm,
                         eps = eps_dbscan,
```

```
                              minPts = minPts_dbscan)$cluster
  fim_dbscan <- Sys.time()
  temp_dbscan <- fim_dbscan - ini_dbscan

  list(
    list(kmeans,
         pam,
         clara,
         agl_ward,
         agl_single,
         agl_complete,
         agl_average,
         agl_centroid,
         diana
         birch_k_medias
         ),
    list(temp_kmeans,
         temp_pam,
         temp_clara,
         temp_ward,
         temp_single,
         temp_complete,
         temp_average,
         temp_centroid,
         temp_diana
         temp_birch
         ))
}

pega_medidas <- function(metodo_resultado, nome_metodo,
                         dados_norm,
                         dados_diss,
                         tempo_resultado){
  data.frame(
    metodo = nome_metodo,
    db_index_cent = clusterSim::index.DB(dados_norm,
                                         metodo_resultado)$DB,
    db_index_med = clusterSim::index.DB(
      dados_norm,
      metodo_resultado,
      d = dados_diss,
      centrotypes = "medoids"
    )$DB,
    dunn_index = clValid::dunn(distance = dados_diss,
                               metodo_resultado),
    silh_index = clusterSim::index.S(dados_diss, metodo_resultado),
    ch1_index = clusterSim::index.G1(dados_norm, metodo_resultado),
```

```
193     ch2_index = clusterSim::index.G1(dados_norm, metodo_resultado,
194                                      d = dados_diss, centrotypes = "medoids
    "),
195     tempo = tempo_resultado
196   )
197 }
198
199 simula_agrupamento <- function(semente,
200                                cenario,
201                                n_k = 50,
202                                k = 7,
203                                n_clusters = 7,
204                                ruido = FALSE,
205                                var_ruido = 3, ...){
206
207   dados_simulados <- cria_dados(semente,
208                                 cenario = cenario,
209                                 n_k = n_k,
210                                 k = k,
211                                 ruido = ruido,
212                                 var_ruido = var_ruido)
213
214   dados_norm <- scale(dados_simulados)
215   dados_dist <- dist(dados_norm)
216
217   agrupamentos <- cria_agrupamentos(dados_simulados,
218                                     n_clusters = n_clusters,
219                                     dados_norm,
220                                     dados_dist, ...)
221
222   ordem_agrupamentos <- c(
223     "k-medias",
224     "pam",
225     "clara",
226     "agl_ward",
227     "agl_single",
228     "agl_complete",
229     "agl_average",
230     "agl_centroid",
231     "diana"
232     "birch"
233   )
234
235   lapply(1:length(agrupamentos[[1]]), function(a) {
236     pega_medidas(agrupamentos[[1]][[a]],
237                  ordem_agrupamentos[a],
238                  dados_norm,
```

```
                  dados_dist,
                  agrupamentos[[2]][[a]])
  }) %>% do.call(rbind.data.frame, .)
}

estudo_simulacao <- function(n_reps, cenario,
                             n_processadores,
                             n_k = 50,
                             k = 7,
                             n_clusters = 7,
                             ruido = FALSE,
                             var_ruido = 3, ...){

  parallel::mclapply(1:n_reps, simula_agrupamento,
                     cenario = cenario,
                     mc.cores = n_processadores,
                     n_k = n_k,
                     k = k, n_clusters = n_clusters,
                     ruido = ruido,
                     var_ruido = var_ruido, ...) %>%
    do.call(rbind.data.frame, .)
}
```

6.12.2 Estudo Resumido

```
library(dplyr)
library(patchwork)
library(ggplot2)

## n = 50

resultado_estudo_1_n050 <- estudo_simulacao(1000,
                                            cenario = 1,
                                            n_processadores = 50,
                                            n_k = 50)
saveRDS(resultado_estudo_1_n050, "resultado_estudo_1_n050.rds")

resultado_estudo_1_ruido_n050 <- estudo_simulacao(1000,
                                                  cenario = 1,
                                                  n_processadores = 50,
                                                  n_k = 50,
                                                  ruido = TRUE,
                                                  var_ruido = 3)
saveRDS(resultado_estudo_1_ruido_n050, "resultado_estudo_1_ruido_n050.rds")


resultado_estudo_1_ruido2_n050 <- estudo_simulacao(1000,
                                                   cenario = 1,
```

```
                                                n_processadores = 50,
                                                n_k = 50,
                                                ruido = TRUE,
                                                var_ruido = 20)
saveRDS(resultado_estudo_1_ruido2_n050, "resultado_estudo_1_ruido2_n050.rds
    ")

resultado_estudo_2_n050 <- estudo_simulacao(1000,
                                            cenario = 2,
                                            n_processadores = 50,
                                            n_k = 50)
saveRDS(resultado_estudo_2_n050, "resultado_estudo_2_n050.rds")



resultado_estudo_2_ruido_n050 <- estudo_simulacao(1000,
                                                  cenario = 2,
                                                  n_processadores = 50,
                                                  n_k = 50,
                                                  ruido = TRUE,
                                                  var_ruido = 3)
saveRDS(resultado_estudo_2_ruido_n050, "resultado_estudo_2_ruido_n050.rds")


resultado_estudo_2_ruido2_n050 <- estudo_simulacao(1000,
                                                   cenario = 2,
                                                   n_processadores = 50,
                                                   n_k = 50,
                                                   ruido = TRUE,
                                                   var_ruido = 20)
saveRDS(resultado_estudo_2_ruido2_n050, "resultado_estudo_2_ruido2_n050.rds
    ")

## n = 100


resultado_estudo_1_n100 <- estudo_simulacao(1000,
                                            cenario = 1,
                                            n_processadores = 50,
                                            n_k = 100)
saveRDS(resultado_estudo_1_n100, "resultado_estudo_1_n100.rds")

resultado_estudo_1_ruido_n100 <- estudo_simulacao(1000,
                                                  cenario = 1,
                                                  n_processadores = 50,
                                                  n_k = 100,
                                                  ruido = TRUE,
```

```
                                                var_ruido = 3)
saveRDS(resultado_estudo_1_ruido_n100, "resultado_estudo_1_ruido_n100.rds")


resultado_estudo_1_ruido2_n100 <- estudo_simulacao(1000,
                                                   cenario = 1,
                                                   n_processadores = 50,
                                                   n_k = 100,
                                                   ruido = TRUE,
                                                   var_ruido = 20)
saveRDS(resultado_estudo_1_ruido2_n100, "resultado_estudo_1_ruido2_n100.rds
    ")

resultado_estudo_2_n100 <- estudo_simulacao(1000,
                                            cenario = 2,
                                            n_processadores = 50,
                                            n_k = 100)
saveRDS(resultado_estudo_2_n100, "resultado_estudo_2_n100.rds")


resultado_estudo_2_ruido_n100 <- estudo_simulacao(1000,
                                                  cenario = 2,
                                                  n_processadores = 50,
                                                  n_k = 100,
                                                  ruido = TRUE,
                                                  var_ruido = 3)
saveRDS(resultado_estudo_2_ruido_n100, "resultado_estudo_2_ruido_n100.rds")


resultado_estudo_2_ruido2_n100 <- estudo_simulacao(1000,
                                                   cenario = 2,
                                                   n_processadores = 50,
                                                   n_k = 100,
                                                   ruido = TRUE,
                                                   var_ruido = 20)
saveRDS(resultado_estudo_2_ruido2_n100, "resultado_estudo_2_ruido2_n100.rds
    ")


## n = 200

resultado_estudo_1_n200 <- estudo_simulacao(1000,
                                            cenario = 1,
                                            n_processadores = 50,
                                            n_k = 200)
saveRDS(resultado_estudo_1_n200, "resultado_estudo_1_n200.rds")
```

```

resultado_estudo_1_ruido_n200 <- estudo_simulacao(1000,
                                                  cenario = 1,
                                                  n_processadores = 50,
                                                  n_k = 200,
                                                  ruido = TRUE,
                                                  var_ruido = 3)
saveRDS(resultado_estudo_1_ruido_n200, "resultado_estudo_1_ruido_n200.rds")


resultado_estudo_1_ruido2_n200 <- estudo_simulacao(1000,
                                                   cenario = 1,
                                                   n_processadores = 50,
                                                   n_k = 200,
                                                   ruido = TRUE,
                                                   var_ruido = 20)
saveRDS(resultado_estudo_1_ruido2_n200, "resultado_estudo_1_ruido2_n200.rds
    ")

resultado_estudo_2_n200 <- estudo_simulacao(1000,
                                            cenario = 2,
                                            n_processadores = 50,
                                            n_k = 200)
saveRDS(resultado_estudo_2_n200, "resultado_estudo_2_n200.rds")



resultado_estudo_2_ruido_n200 <- estudo_simulacao(1000,
                                                  cenario = 2,
                                                  n_processadores = 50,
                                                  n_k = 200,
                                                  ruido = TRUE,
                                                  var_ruido = 3)
saveRDS(resultado_estudo_2_ruido_n200, "resultado_estudo_2_ruido_n200.rds")


resultado_estudo_2_ruido2_n200 <- estudo_simulacao(1000,
                                                   cenario = 2,
                                                   n_processadores = 50,
                                                   n_k = 200,
                                                   ruido = TRUE,
                                                   var_ruido = 20)
saveRDS(resultado_estudo_2_ruido2_n200, "resultado_estudo_2_ruido2_n200.rds
    ")
```

6.12.3 Estudo Resumido - diferentes valores de K

```
library(dplyr)
```

```r
library(patchwork)
library(ggplot2)

n_clusters_l <- 5:9
cenarios <- c(1, 2)
n_sample <- c(50)

resultado_estudo_l <- lapply(cenarios, function(cen_tipo){
  lapply(n_sample, function(n_size){
    lapply(n_clusters_l, function(n_clust){
      temp_data <- estudo_simulacao(1000,
                                    cenario = cen_tipo,
                                    n_processadores = 6,
                                    n_k = n_size,
                                    k = 7,
                                    n_clusters = n_clust)
      temp_data$n_clusters = n_clust
      temp_data$n_k = n_size
      temp_data$cenario = cen_tipo
      temp_data$ruidos = "00 Var. Ruido"
      temp_data
    }) %>% do.call(rbind.data.frame, .)
  }) %>% do.call(rbind.data.frame, .)
}) %>% do.call(rbind.data.frame, .)

saveRDS(resultado_estudo_l,
        "resultado_estudo_l.rds")

resultado_estudo_l_ruido1 <- lapply(cenarios, function(cen_tipo){
  lapply(n_sample, function(n_size){
    lapply(n_clusters_l, function(n_clust){
      temp_data <- estudo_simulacao(1000,
                                    cenario = cen_tipo,
                                    n_processadores = 6,
                                    n_k = n_size,
                                    k = 7,
                                    n_clusters = n_clust,
                                    ruido = TRUE,
                                    var_ruido = 3)
      temp_data$n_clusters = n_clust
      temp_data$n_k = n_size
      temp_data$cenario = cen_tipo
      temp_data$ruidos = "03 Var. Ruido"
      temp_data
    }) %>% do.call(rbind.data.frame, .)
  }) %>% do.call(rbind.data.frame, .)
}) %>% do.call(rbind.data.frame, .)
```

```
49
50 saveRDS(resultado_estudo_l_ruido1,
51         "resultado_estudo_l_ruido1.rds")
52
53 resultado_estudo_l_ruido2 <- lapply(cenarios, function(cen_tipo){
54   lapply(n_sample, function(n_size){
55     lapply(n_clusters_l, function(n_clust){
56       temp_data <- estudo_simulacao(1000,
57                                     cenario = cen_tipo,
58                                     n_processadores = 6,
59                                     n_k = n_size,
60                                     k = 7,
61                                     n_clusters = n_clust,
62                                     ruido = TRUE,
63                                     var_ruido = 20)
64       temp_data$n_clusters = n_clust
65       temp_data$n_k = n_size
66       temp_data$cenario = cen_tipo
67       temp_data$ruidos = "20 Var. Ruido"
68       temp_data
69     }) %>% do.call(rbind.data.frame, .)
70   }) %>% do.call(rbind.data.frame, .)
71 }) %>% do.call(rbind.data.frame, .)
72
73 saveRDS(resultado_estudo_l_ruido2,
74         "resultado_estudo_l_ruido2.rds")
75
76 listas_resultados <- list(resultado_estudo_l,
77                           resultado_estudo_l_ruido1,
78                           resultado_estudo_l_ruido2)
79
80 listas_resultados <- do.call(rbind.data.frame,
81                              listas_resultados)
82
83 listas_resultados$cenario <- paste0("Cenario ",
84                                     listas_resultados$cenario)
85
86 listas_resultados$ruidos2 <-
87   ifelse(listas_resultados$ruidos == "00 Var. Ruido",
88          "00 Ruidos",
89          ifelse(listas_resultados$ruidos == "03 Var. Ruido",
90                 "03 Ruidos", "20 Ruidos"))
91
92
93 compara_n_clusters <- function(metodo_escolhido){
94   g <- listas_resultados %>%
95     filter(metodo == metodo_escolhido) %>%
```

```
    ggplot() +
    theme_minimal() +
    aes(x = factor(n_clusters), fill = factor(n_clusters)) +
    geom_boxplot() +
    facet_grid(ruidos2 ~ cenario, scales = "free") +
    labs(x = "N mero de clusters") +
    scale_fill_viridis_d() +
    theme(legend.position = "none")

  g1 <- g +
    aes(y = ch1_index) +
    labs(y = "Indice CH")

  g2 <- g +
    aes(y = silh_index) +
    labs(y = "Indice Silhueta")

  g3 <- g +
    aes(y = silh_index) +
    labs(y = "Indice Dunn")

  g4 <- g +
    aes(y = silh_index) +
    labs(y = "Indice Davies-Bouldin")

  (g1 + g2)/(g3 + g4)
}


png("sim_k-medias.png")
compara_n_clusters("k-medias")
dev.off()

png("sim_single.png")
compara_n_clusters("agl_single")
dev.off()

png("sim_complete.png")
compara_n_clusters("agl_complete")
dev.off()

png("sim_centroid.png")
compara_n_clusters("agl_centroid")
dev.off()

png("sim_diana.png")
compara_n_clusters("diana")
```

```
143 dev.off()
144
145
146 listas_resumo <- listas_resultados %>%
147   group_by(metodo, cenario, ruidos2, n_clusters) %>%
148   summarise(media_db_index_cent = mean(db_index_cent),
149             sd_db_index_cent = sd(db_index_cent),
150             media_db_index_med = mean(db_index_med),
151             sd_db_index_med = sd(db_index_med),
152             media_dunn_index = mean(dunn_index),
153             sd_dunn_index = sd(dunn_index),
154             media_silh_index = mean(silh_index),
155             sd_silh_index = sd(silh_index),
156             media_ch1_index = mean(ch1_index),
157             sd_ch1_index = sd(ch1_index),
158             media_ch2_index = mean(ch2_index),
159             sd_ch2_index = sd(ch2_index))
160
161 ## Considerando somente valores m dios
162 g <- ggplot(listas_resumo) +
163   theme_minimal() +
164   aes(color = metodo, x = n_clusters) +
165   scale_color_viridis_d() +
166   facet_grid(ruidos2 ~ cenario, scales = "free") +
167   labs(x = "N mero de clusters")
168
169
170 g1 <- g + geom_line(aes(y = media_ch1_index)) +
171   labs(y = "Media Indice CH")
172
173 g2 <- g + geom_line(aes(y = media_dunn_index)) +
174   labs(y = "Media Indice Dunn")
175
176 g3 <- g + geom_line(aes(y = media_db_index_cent)) +
177   labs(y = "Media Indice Davies-Bouldin")
178
179 g4 <- g + geom_line(aes(y = media_silh_index)) +
180   labs(y = "Media Indice Silhueta")
181
182 png("simulacao_compara_k.png")
183 g1 + g2 + g3 + g4 +
184   plot_layout(guides = 'collect') & theme(legend.position = 'bottom')
185 dev.off()
```

Referências

Ankerst, M., Breunig, M. M., peter Kriegel, H., e Sander, J. (1999), "OPTICS: Ordering Points To Identify the Clustering Structure," in *Proc. ACM SIGMOD Int. Conf. on Management of Data*, ACM Press, pp. 49–60. Citado na página 29.

Boyko, N., Hetman, S., e Kots, I. (2021), "Comparison of Clustering Algorithms for Revenue and Cost Analysis," in *Proceedings of the 5th International Conference on Computational Linguistics and Intelligent Systems (COLINS 2021). Volume I: Main Conference, Lviv, Ukraine, April 22-23, 2021*, eds. Sharonova, N., Lytvyn, V., Cherednichenko, O., Kupriianov, Y., Kanishcheva, O., Hamon, T., Grabar, N., Vysotska, V., Kowalska-Styczen, A., e Jonek-Kowalska, I., CEUR-WS.org, vol. 2870 of *CEUR Workshop Proceedings*, pp. 1866–1877. Citado na página 26.

Brock, G., Pihur, V., Datta, S., e Datta, S. (2008), "clValid: An R Package for Cluster Validation," *Journal of Statistical Software*, 25, 1–22. Citado na página 55.

Caliński, T. e Harabasz, J. (1974), "A dendrite method for cluster analysis," *Communications in Statistics*, 3, 1–27. Citado na página 33.

Davies, D. L. e Bouldin, D. W. (1979), "A Cluster Separation Measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1, 224–227. Citado na página 31.

Dunn, J. C. (1974), "Well-Separated Clusters and Optimal Fuzzy Partitions," *Journal of Cybernetics*, 4, 95–104. Citado na página 32.

Ester, M., Kriegel, H.-P., Sander, J., e Xu, X. (1996), "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, AAAI Press, pp. 226–231. Citado na página 26.

Filippone, M., Camastra, F., Masulli, F., e Rovetta, S. (2008), "A survey of kernel and spectral methods for clustering," *Pattern Recognition*, 41, 176–190. Citado na página 29.

Guha, S., Rastogi, R., e Shim, K. (2000), "Rock: A robust clustering algorithm for categorical attributes," *Information Systems*, 25, 345–366. Citado na página 26.

Hahsler, M. e Piekenbrock, M. (2022), *dbscan: Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Related Algorithms*, r package version 1.1-10. Citado na página 49.

Halkidi, M., Batistakis, Y., e Vazirgiannis, M. (2001), "On Clustering Validation Techniques," *Journal of Intelligent Information Systems*, 17, 107–145. Citado na página 31.

Han, J., Kamber, M., e Pei, J. (2012), *Data Mining: Concepts and Techniques*, The Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann, third edition ed. Citado 2 vezes nas páginas 25 e 29.

Hartigan, J. A. e Wong, M. A. (1979), "Algorithm AS 136: A K-Means Clustering Algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28, 100–108. Citado 2 vezes nas páginas 19 e 44.

Jorge, M. H. P. d. M., Laurenti, R., e Gotlieb, S. L. D. (2007), "[Quality analysis of Brazilian vital statistics: the experience of implementing the SIM and SINASC systems]," *Ciencia & saude coletiva*, 12, 643–654. Citado na página 12.

Karypis, G., Han, E.-H., e Kumar, V. (1999), "Chameleon: hierarchical clustering using dynamic modeling," *Computer*, 32, 68–75. Citado na página 25.

Kassambara, A. e Mundt, F. (2020), *factoextra: Extract and Visualize the Results of Multivariate Data Analyses*, r package version 1.0.7. Citado na página 44.

Kaufman, L. e Rousseeuw, P. J. (1990), *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley. Citado 2 vezes nas páginas 20 e 22.

Liu, Y., Li, Z., Xiong, H., Gao, X., e Wu, J. (2010), "Understanding of Internal Clustering Validation Measures," in *2010 IEEE International Conference on Data Mining*, pp. 911–916. Citado na página 31.

Lloyd, S. (1982), "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, 28, 129–137. Citado na página 19.

Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M., e Hornik, K. (2021), *cluster: Cluster Analysis Basics and Extensions*, r package version 2.1.2 — For new features, see the 'Changelog' file (in the package source). Citado na página 44.

Nasteski, V. (2017), "An overview of the supervised machine learning methods," *HORIZONS.B*, 4, 51–62. Citado na página 13.

Ng, A. Y., Jordan, M. I., e Weiss, Y. (2001), "On Spectral Clustering: Analysis and an Algorithm," in *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, Cambridge, MA, USA: MIT Press, NIPS'01, p. 849–856. Citado na página 30.

R Core Team (2021), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria. Citado 2 vezes nas páginas 14 e 44.

Sander, J., Ester, M., Kriegel, H.-P., e Xu, X. (2004), "Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications," *Data Mining and Knowledge Discovery*, 2, 169–194. Citado na página 28.

Therneau, T. e Atkinson, B. (2019), *rpart: Recursive Partitioning and Regression Trees*, r package version 4.1-15. Citado na página 60.

Verma, D. e Meila, M. (2003), "A Comparison of Spectral Clustering Algorithms," Tech. rep., University of Washington. Citado na página 30.

Walesiak, M. e Dudek, A. (2020), "The Choice of Variable Normalization Method in Cluster Analysis," in *Education Excellence and Innovation Management: A 2025 Vision to Sustain Economic Development During Global Challenges*, ed. Soliman, K. S., International Business Information Management Association (IBIMA), pp. 325–340. Citado na página 55.

Ward, J. H. (1963), "Hierarchical Grouping to Optimize an Objective Function," *Journal of the American Statistical Association*, 58, 236–244. Citado na página 22.

Wei, T. e Simko, V. (2017), *R package "corrplot": Visualization of a Correlation Matrix*, (Version 0.84). Citado na página 42.

Xu, D. e Tian, Y. (2015), "A Comprehensive Survey of Clustering Algorithms," *Annals of Data Science*, 2, 165–193. Citado na página 13.

Zelnik-manor, L. e Perona, P. (2004), "Self-Tuning Spectral Clustering," in *Advances in Neural Information Processing Systems*, eds. Saul, L., Weiss, Y., e Bottou, L., MIT Press, vol. 17. Citado na página 30.

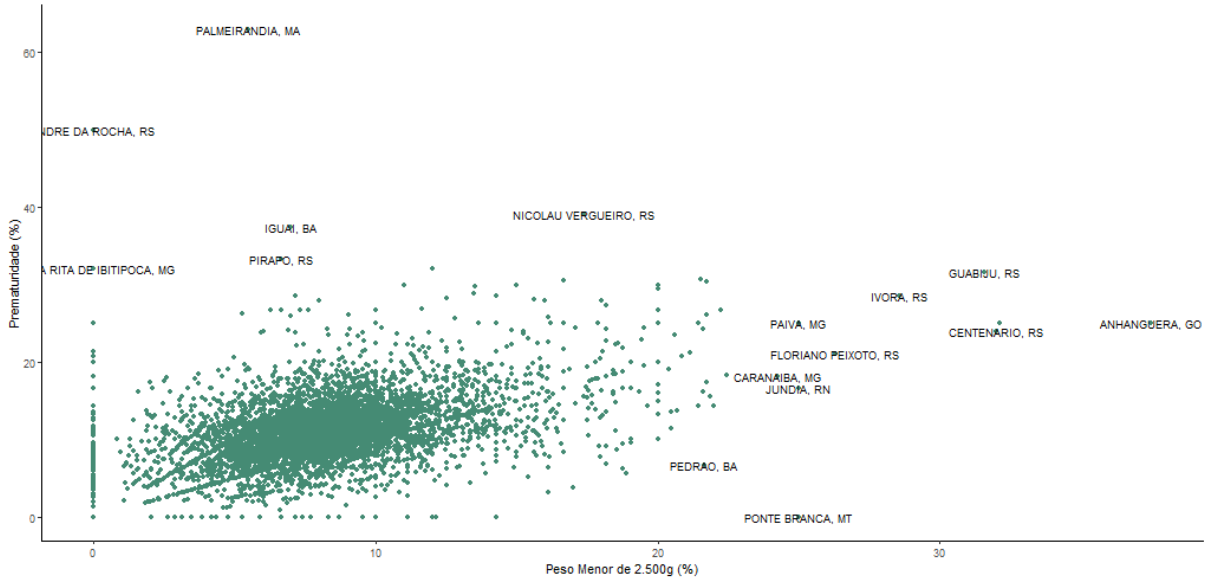
Zhang, T., Ramakrishnan, R., e Livny, M. (1996), "BIRCH: An Efficient Data Clustering Method for Very Large Databases," in *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA: Association for Computing Machinery, SIGMOD '96, p. 103–114. Citado na página 23.

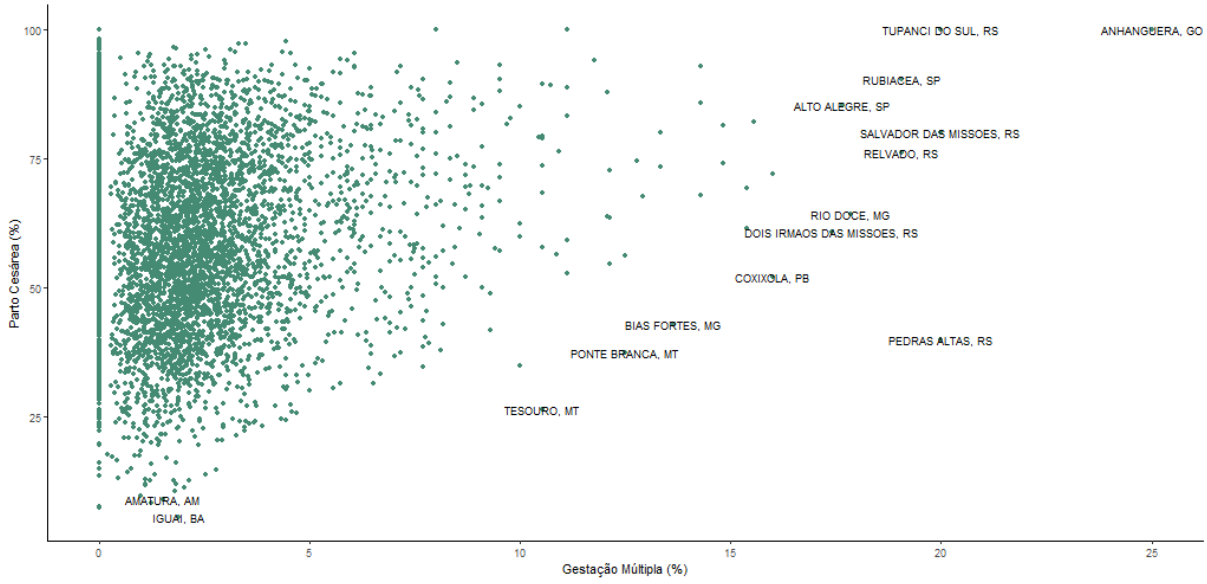
Apêndices

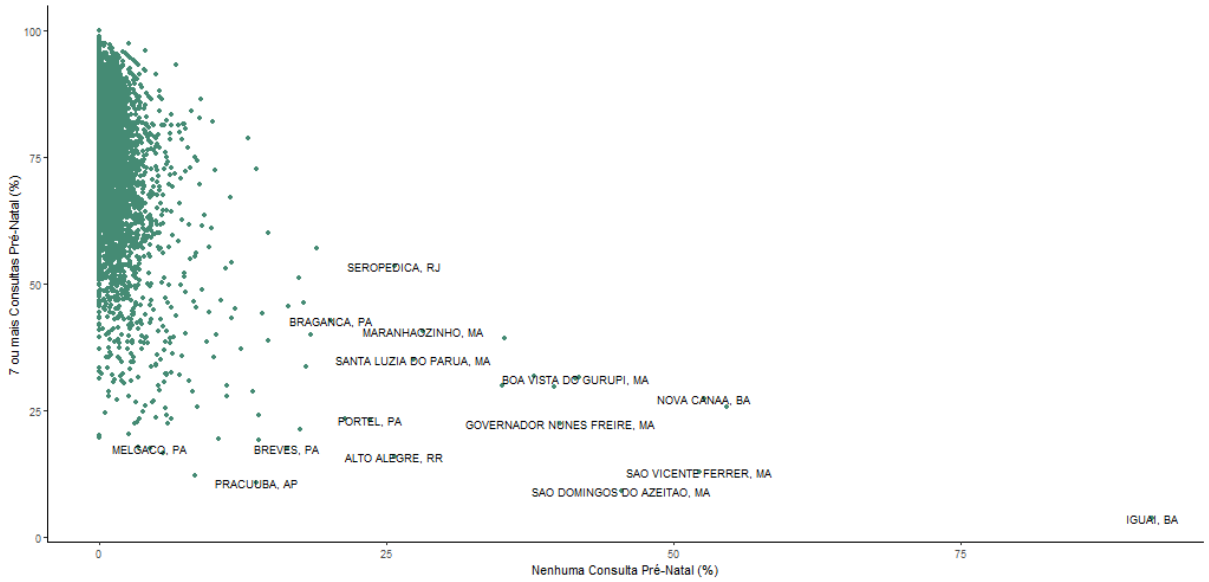
APÊNDICE A – Municípios ignorados

uf	codigo	nascidos_vivos	porc_premat	porc_gesta_multipla	porc_cesarea	porc_0_consulta	porc_7mais_consulta	porc_apgar1_menor_7	porc_apgar5_menor_7	porc_anomalia	porc_peso_menor_2500	porc_fem
AC	120000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
AL	270000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
AM	130000	12	18,18	0	50,00	18,18	27,27	0,00	0,00	0	8,33	41,67
BA	290000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
CE	230000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
ES	320000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
GO	520000	4	50,00	0	0,00	25,00	50,00	33,33	33,33	0	50,00	100,00
MA	210000	8	0,00	0	75,00	25,00	25,00	0,00	0,00	0	NA	75,00
MG	310000	2	0,00	0	50,00	0,00	100,00	0,00	0,00	0	50,00	0,00
MS	500000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
MT	510000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
PA	150000	3	0,00	0	66,67	33,33	33,33	0,00	0,00	0	NA	100,00
PB	250000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
PE	260000	13	0,00	0	53,85	9,09	81,82	0,00	0,00	0	NA	53,85
PI	220000	2	0,00	0	50,00	0,00	100,00	0,00	0,00	0	NA	100,00
PR	410000	1	0,00	NA	NA	100,00	0,00	NA	NA	0	100,00	100,00
RJ	330000	23	15,79	0	5,00	22,22	61,11	5,88	5,88	0	17,39	47,83
RN	240000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
RO	110000	1	100,00	0	0,00	100,00	0,00	0,00	0,00	0	NA	100,00
SC	420000	3	0,00	0	33,33	33,33	33,33	0,00	0,00	0	NA	66,67
SE	280000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
SP	350000	8	20,00	0	33,33	33,33	33,33	20,00	0,00	0	25,00	75,00
TO	170000	2	0,00	100	100,00	0,00	100,00	0,00	0,00	0	NA	50,00

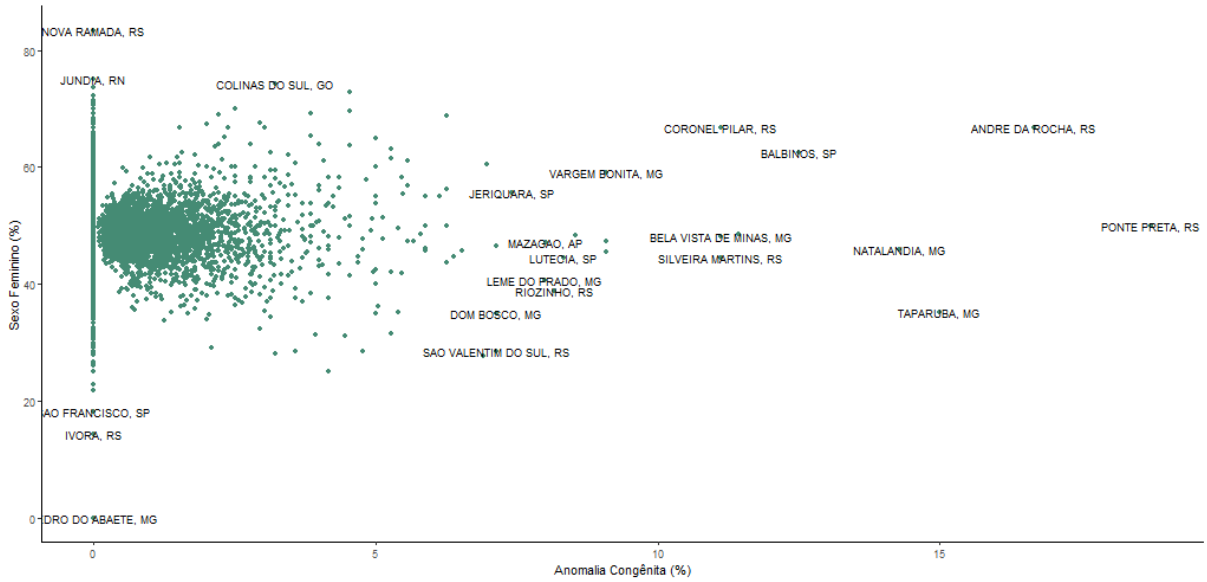
APÊNDICE B – Gráficos dos indicadores obstétricos



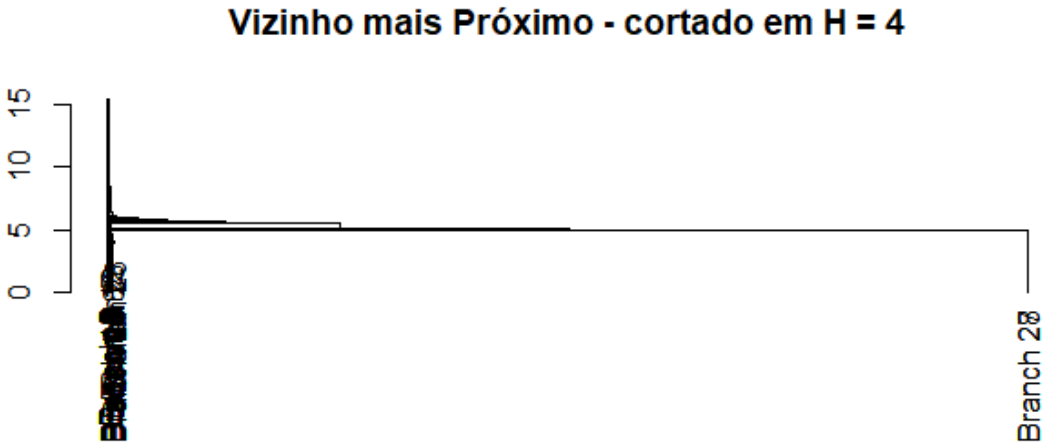


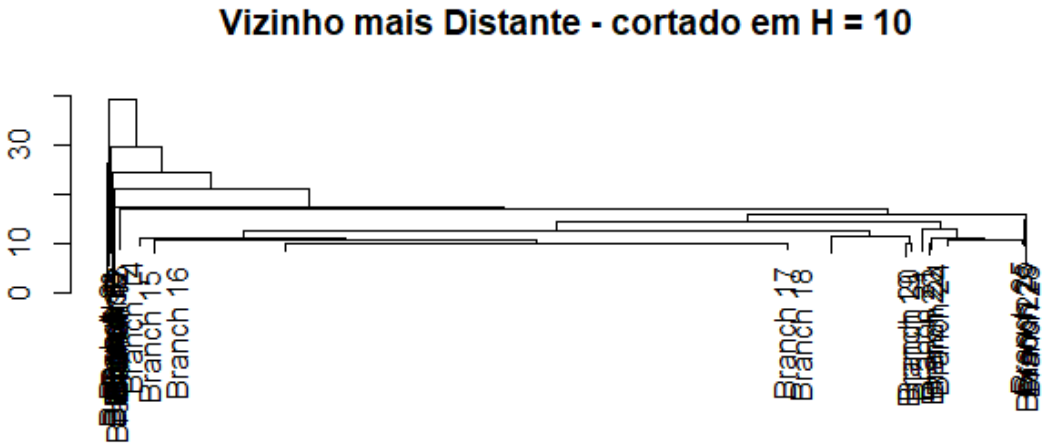

7 ou mais Consultas Pré-Natal (%)
Nenhuma Consulta Pré-Natal (%)
SEROPEDICA, RJ
BRAGANCA, PA
MARANHAOZINHO, MA
SANTA LUZIA DO PARUA, MA
BOA VISTA DO GURUPI, MA
NOVA CANAA, BA
PORTEL, PA
GOVERNADOR NUNES FREIRE, MA
MELGACO, PA
BREVES, PA
ALTO ALEGRE, RR
SAO VICENTE FERRER, MA
PRACUUBA, AP
SAO DOMINGOS DO AZEITAO, MA
IGUAI, BA

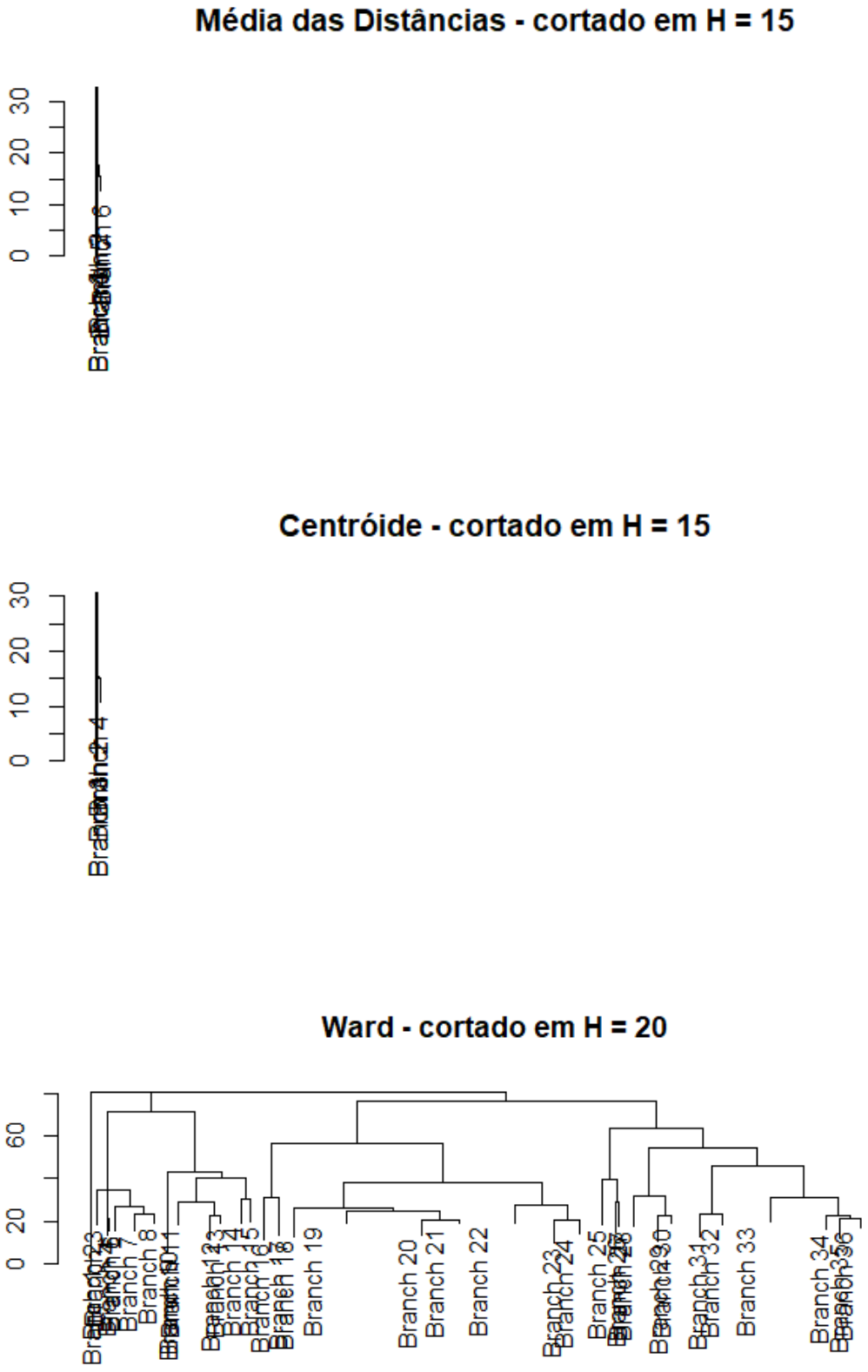

Apgar do 5º Minuto menor que 7 (%)
Apgar do 1º Minuto menor que 7 (%)
LAGOA DO BARRO DO PIAUI, PI
GUARIBAS, PI
CARACOL, PI
DOM INOCENCIO, PI
FARTURA DO PIAUI, PI
DIRCEU ARCOVERDE, PI
RUBELITA, MG
BURITI BRAVO, MA
VITORIA DO MEARIM, MA
ACREUNA, GO
SANTA CRUZ DO ARARI
GOVERNADOR EUGENIO BARROS, MA

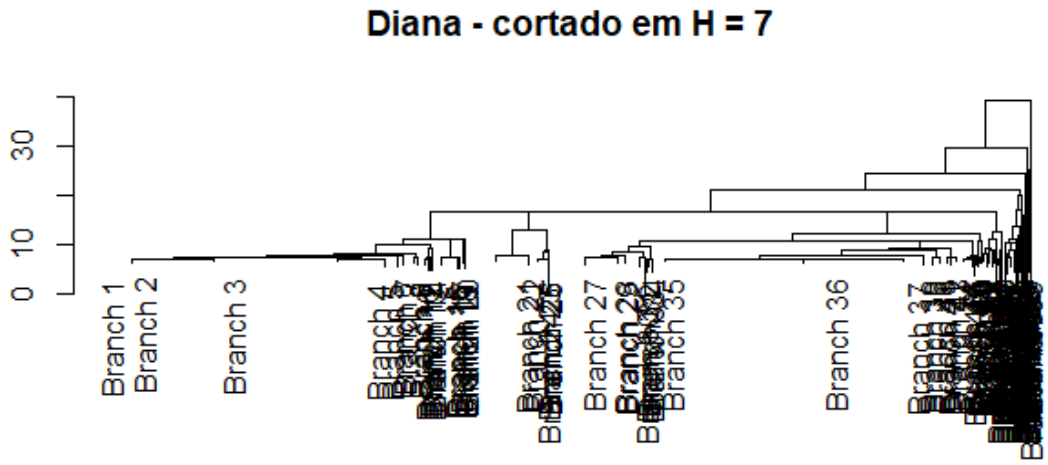

NOVA RAMADA, RS
JUNDIÁ, RN
COLINAS DO SUL, GO
CORONEL PILAR, RS
ANDRE DA ROCHA, RS
BALBINOS, SP
VARGEM BONITA, MG
JERIQUARA, SP
PONTE PRETA, RS
MAZAGÃO, AP
BELA VISTA DE MINAS, MG
NATALÂNDIA, MG
LUTECIA, SP
SILVEIRA MARTINS, RS
LEME DO PRADO, MG
RIOZINHO, RS
DOM BOSCO, MG
TAPARUBA, MG
SAO VALENTIM DO SUL, RS
SÃO FRANCISCO, SP
IVORÁ, RS
DRO DO ABAETE, MG
Sexo Feminino (%)
Anomalia Congênita (%)
0
20
40
60
80
0
5
10
15

APÊNDICE C – Dendrogramas dos métodos hierárquicos






Média das Distâncias - cortado em H = 15
0 10 20 30
Centróide - cortado em H = 15
0 10 20 30
Ward - cortado em H = 20
0 20 60
Branch 19
Branch 20
Branch 21
Branch 22
Branch 23
Branch 24
Branch 25
Branch 31
Branch 32
Branch 33
Branch 34


Diana - cortado em H = 7
0
10
30
Branch 1
Branch 2
Branch 3
Branch 27
Branch 35
Branch 36
Branch 37

APÊNDICE D – Municípios identificados como *outliers* pelo método de agrupamento DBSCAN

Município - UF			
PARECIS - RO	BARRO PRETO - BA	SERRA DA SAUDADE - MG	CORONEL PILAR - RS
SANTA ROSA DO PURUS - AC	CATOLANDIA - BA	SILVEIRANIA - MG	COTIPORA - RS
AMATURA - AM	FEIRA DA MATA - BA	TAPARUBA - MG	CRISSIUMAL - RS
CANUTAMA - AM	GAVIAO - BA	TAQUARACU DE MINAS - MG	DEZESSEIS DE NOVEMBRO - RS
CARAUARI - AM	IGUAI - BA	UMBURATIBA - MG	DOIS IRMAOS DAS MISSOES - RS
SAO PAULO DE OLIVENCA - AM	IPIRA - BA	VARGEM BONITA - MG	ENTRE RIOS DO SUL - RS
ALTO ALEGRE - RR	IPUPIARA - BA	VEREDINHA - MG	ERNESTINA - RS
AMAJARI - RR	ITAQUARA - BA	VERISSIMO - MG	ESMERALDA - RS
PACARAIMA - RR	LAMARAO - BA	LAJE DO MURIAE - RJ	EUGENIO DE CASTRO - RS
PEIXE-BOI - PA	NOVA CANAA - BA	SANTA MARIA MADALENA - RJ	FAZENDA VILANOVA - RS
PORTEL - PA	NOVA IBIA - BA	SEROPEDICA - RJ	FLORIANO PEIXOTO - RS
SANTA CRUZ DO ARARI - PA	PEDRAO - BA	ALFREDO MARCONDES - SP	FORMIGUEIRO - RS
CALCOENE - AP	POTIRAGUA - BA	ALTO ALEGRE - SP	GARRUCHOS - RS
ITAUBAL - AP	RIBEIRA DO AMPARO - BA	ALVARO DE CARVALHO - SP	GRAMADO DOS LOUREIROS - RS
MAZAGAO - AP	RIBEIRAO DO LARGO - BA	ALVINLANDIA - SP	GRAMADO XAVIER - RS
PRACUUBA - AP	SALINAS DA MARGARIDA - BA	BALBINOS - SP	GUABIJU - RS
SANTANA - AP	SANTA CRUZ DA VITORIA - BA	BARAO DE ANTONINA - SP	GUARANI DAS MISSOES - RS
SERRA DO NAVIO - AP	SANTANOPOLIS - BA	BORACEIA - SP	ILOPOLIS - RS
TARTARUGALZINHO - AP	SAUBARA - BA	CANDIDO RODRIGUES - SP	ITACURUBI - RS
ANGICO - TO	SERRA PRETA - BA	CORUMBATAI - SP	ITAPUCA - RS
CENTENARIO - TO	SERRINHA - BA	ELISIARIO - SP	ITATIBA DO SUL - RS
LAGOA DO TOCANTINS - TO	TEODORO SAMPAIO - BA	FERNAO - SP	IVORA - RS
LAVANDEIRA - TO	URUCUCA - BA	FLORA RICA - SP	JARI - RS
NOVO JARDIM - TO	VEREDA - BA	GALIA - SP	LAGOAO - RS
OLIVEIRA DE FATIMA - TO	ABADIA DOS DOURADOS - MG	GASTAO VIDIGAL - SP	LINHA NOVA - RS
PONTE ALTA DO BOM JESUS - TO	AGUA COMPRIDA - MG	GETULINA - SP	MANOEL VIANA - RS
PORTO ALEGRE DO TOCANTINS - TO	ALAGOA - MG	INUBIA PAULISTA - SP	MAQUINE - RS
SANTA ROSA DO TOCANTINS - TO	ALBERTINA - MG	IPAUSSU - SP	MATA - RS
SAO FELIX DO TOCANTINS - TO	ALMENARA - MG	JERIQUARA - SP	MATO CASTELHANO - RS
SAO SALVADOR DO TOCANTINS - TO	ALVORADA DE MINAS - MG	LOURDES - SP	MATO QUEIMADO - RS
SUCUPIRA - TO	BANDEIRA - MG	LUTECIA - SP	MONTAURI - RS
TAIPAS DO TOCANTINS - TO	BARRA LONGA - MG	MARINOPOLIS - SP	MORRINHOS DO SUL - RS
TUPIRAMA - TO	BELA VISTA DE MINAS - MG	MIRA ESTRELA - SP	NICOLAU VERGUEIRO - RS
ARAIOSES - MA	BELMIRO BRAGA - MG	PARANAPUA - SP	NOVA BOA VISTA - RS
BURITI BRAVO - MA	BIAS FORTES - MG	POLONI - SP	NOVA BRESCIA - RS
CAJARI - MA	BONFIM - MG	PONTALINDA - SP	NOVA RAMADA - RS
CENTRO DO GUILHERME - MA	CAMPANARIO - MG	PRACINHA - SP	NOVA ROMA DO SUL - RS
CIDELANDIA - MA	CAMPO AZUL - MG	QUINTANA - SP	NOVO XINGU - RS
GOVERNADOR EUGENIO BARROS - MA	CARANAIBA - MG	RIFAINA - SP	PAIM FILHO - RS
MARACACUME - MA	CARMESIA - MG	RUBIACEA - SP	PARECI NOVO - RS
MARANHAOZINHO - MA	CARRANCAS - MG	SANTA CRUZ DA ESPERANCA - SP	PAULO BENTO - RS
NOVA OLINDA DO MARANHAO - MA	CATUTI - MG	SANTA RITA D'OESTE - SP	PEDRAS ALTAS - RS
PALMEIRANDIA - MA	CEDRO DO ABAETE - MG	SANTA SALETE - SP	PINHAL - RS
PERI MIRIM - MA	CHACARA - MG	SAO FRANCISCO - SP	PINHAL DA SERRA - RS
PRESIDENTE MEDICI - MA	CLARAVAL - MG	SUZANAPOLIS - SP	PINHEIRINHO DO VALE - RS
SAMBAIBA - MA	COMERCINHO - MG	UNIAO PAULISTA - SP	PIRAPO - RS
SANTA LUZIA DO PARUA - MA	CONCEICAO DAS PEDRAS - MG	ALTAMIRA DO PARANA - PR	POCO DAS ANTAS - RS
SAO DOMINGOS DO AZEITAO - MA	CONSOLACAO - MG	ANAHY - PR	PONTE PRETA - RS
SAO JOAO BATISTA - MA	CORDISLANDIA - MG	ARIRANHA DO IVAI - PR	PORTO MAUA - RS
SAO LUIS GONZAGA DO MARANHAO - MA	CORREGO NOVO - MG	CAMPO BONITO - PR	PROTASIO ALVES - RS
SAO VICENTE FERRER - MA	CRUCILANDIA - MG	CAPITAO LEONIDAS MARQUES - PR	QUATRO IRMAOS - RS
TURILANDIA - MA	DOM BOSCO - MG	GODOY MOREIRA - PR	RELVADO - RS
VITORIA DO MEARIM - MA	DOM JOAQUIM - MG	IGUATU - PR	RIOZINHO - RS
BELA VISTA DO PIAUI - PI	DONA EUSEBIA - MG	IRACEMA DO OESTE - PR	SALVADOR DAS MISSOES - RS
CARACOL - PI	DORESOPOLIS - MG	JANIOPOLIS - PR	SANTA MARGARIDA DO SUL - RS
COCAL DE TELHA - PI	FAMA - MG	JAPIRA - PR	SANTANA DA BOA VISTA - RS

Município - UF			
CRISTALANDIA DO PIAUI - PI	FELISBURGO - MG	JARDIM OLINDA - PR	SANTO ANTONIO DO PLANALTO - RS
DIRCEU ARCOVERDE - PI	FERNANDES TOURINHO - MG	KALORE - PR	SAO JOAO DO POLESINE - RS
DOM INOCENCIO - PI	FRUTA DE LEITE - MG	MARUMBI - PR	SAO JORGE - RS
ELISEU MARTINS - PI	GAMELEIRAS - MG	NOVA OLIMPIA - PR	SAO PEDRO DAS MISSOES - RS
FARTURA DO PIAUI - PI	GOIANA - MG	NOVO ITACOLOMI - PR	SAO VALENTIM - RS
FLORES DO PIAUI - PI	GONCALVES - MG	PAULO FRONTIN - PR	SAO VALENTIM DO SUL - RS
FLORESTA DO PIAUI - PI	INHAUMA - MG	PRADO FERREIRA - PR	SAO VALERIO DO SUL - RS
FRANCISCO MACEDO - PI	ITACAMBIRA - MG	PRESIDENTE CASTELO BRANCO - PR	SEVERIANO DE ALMEIDA - RS
GUARIBAS - PI	ITAMARATI DE MINAS - MG	RANCHO ALEGRE D'OESTE - PR	SILVEIRA MARTINS - RS
JARDIM DO MULATO - PI	ITAMBE DO MATO DENTRO - MG	SANTA CECILIA DO PAVAO - PR	TIO HUGO - RS
JERUMENHA - PI	JACINTO - MG	SANTA CRUZ DE MONTE CASTELO - PR	TIRADENTES DO SUL - RS
LAGOA DO BARRO DO PIAUI - PI	JOANESIA - MG	UNIFLOR - PR	TOROPI - RS
PAES LANDIM - PI	JURAMENTO - MG	ATALANTA - SC	TRES ARROIOS - RS
PAJEU DO PIAUI - PI	JUVENILIA - MG	BARRA BONITA - SC	TUNAS - RS
PAU D'ARCO DO PIAUI - PI	LAGAMAR - MG	BELMONTE - SC	TUPANCI DO SUL - RS
PEDRO LAURENTINO - PI	LARANJAL - MG	BOCAINA DO SUL - SC	UBIRETAMA - RS
RIACHO FRIO - PI	LEME DO PRADO - MG	BOM JARDIM DA SERRA - SC	UNIAO DA SERRA - RS
SAO FELIX DO PIAUI - PI	MARAVILHAS - MG	BOM RETIRO - SC	VANINI - RS
SAO FRANCISCO DE ASSIS DO PIAUI - PI	MONJOLOS - MG	BRUNOPOLIS - SC	VESPASIANO CORREA - RS
SAO JOAO DA CANABRAVA - PI	MONTALVANIA - MG	CAPAO ALTO - SC	VILA FLORES - RS
SAO JOAO DA VARJOTA - PI	MONTE AZUL - MG	CORDILHEIRA ALTA - SC	VILA MARIA - RS
VILA NOVA DO PIAUI - PI	MORRO DA GARCA - MG	ERVAL VELHO - SC	VILA NOVA DO SUL - RS
WALL FERRAZ - PI	MORRO DO PILAR - MG	GALVAO - SC	ALCINOPOLIS - MS
ERERE - CE	MUNHOZ - MG	IBIAM - SC	JAPORA - MS
AGUA NOVA - RN	NACIP RAYDAN - MG	LACERDOPOLIS - SC	MUNDO NOVO - MS
FRANCISCO DANTAS - RN	NATALANDIA - MG	LAJEADO GRANDE - SC	TACURU - MS
IPUEIRA - RN	OLARIA - MG	LEOBERTO LEAL - SC	CAMPINAPOLIS - MT
JUNDIA - RN	ONCA DE PITANGUI - MG	LUZERNA - SC	FIGUEIROPOLIS D'OESTE - MT
MAJOR SALES - RN	ORATORIOS - MG	MAJOR GERCINO - SC	NOVA NAZARE - MT
OLHO-D'AGUA DO BORGES - RN	PAIVA - MG	MODELO - SC	PONTE BRANCA - MT
OURO BRANCO - RN	PARACATU - MG	OURO VERDE - SC	TESOURO - MT
RUY BARBOSA - RN	PEDRA DO ANTA - MG	PALMEIRA - SC	UNIAO DO SUL - MT
SAO VICENTE - RN	PEDRINOPOLIS - MG	PERITIBA - SC	ACREUNA - GO
SERRINHA - RN	PEDRO TEIXEIRA - MG	PINHEIRO PRETO - SC	ADELANDIA - GO
AMPARO - PB	PEQUI - MG	SAO BONIFACIO - SC	AGUA FRIA DE GOIAS - GO
BOM JESUS - PB	PERIQUITO - MG	SUL BRASIL - SC	ALVORADA DO NORTE - GO
BOM SUCESSO - PB	PIAU - MG	URUPEMA - SC	ANHANGUERA - GO
COXIXOLA - PB	PRATINHA - MG	ALECRIM - RS	ARUANA - GO
FREI MARTINHO - PB	QUARTEL GERAL - MG	ALTO FELIZ - RS	AVELINOPOLIS - GO
LAGOA - PB	RIO DOCE - MG	ANDRE DA ROCHA - RS	COLINAS DO SUL - GO
MATINHAS - PB	RUBELITA - MG	ARAMBARE - RS	DAVINOPOLIS - GO
MATO GROSSO - PB	SALTO DA DIVISA - MG	ARROIO DO PADRE - RS	EDEALINA - GO
PASSAGEM - PB	SANTA BARBARA DO MONTE VERDE - MG	BARAO DO TRIUNFO - RS	FAZENDA NOVA - GO
PEDRA LAVRADA - PB	SANTA FE DE MINAS - MG	BARRA DO QUARAI - RS	GUARANI DE GOIAS - GO
POCO DE JOSE DE MOURA - PB	SANTA RITA DE IBITIPOCA - MG	BARRA FUNDA - RS	IPIRANGA DE GOIAS - GO
SANTA INES - PB	SANTO ANTONIO DO MONTE - MG	BENJAMIN CONSTANT DO SUL - RS	JESUPOLIS - GO
SANTAREM - PB	SANTO HIPOLITO - MG	BOZANO - RS	LAGOA SANTA - GO
SERRA DA RAIZ - PB	SAO FELIX DE MINAS - MG	BROCHIER - RS	MAMBAI - GO
SERRA GRANDE - PB	SAO GERALDO DA PIEDADE - MG	CAIBATE - RS	MORRO AGUDO DE GOIAS - GO
TRIUNFO - PB	SAO GONCALO DO ABAETE - MG	CAMPESTRE DA SERRA - RS	NOVA IGUACU DE GOIAS - GO
ZABELE - PB	SAO GONCALO DO RIO PRETO - MG	CAMPINA DAS MISSOES - RS	PALESTINA DE GOIAS - GO
ITACURUBA - PE	SAO JOSE DO JACURI - MG	CARLOS GOMES - RS	PALMELO - GO
PINDOBA - AL	SAO SEBASTIAO DO MARANHAO - MG	CENTENARIO - RS	SANTA RITA DO NOVO DESTINO - GO
AMPARO DE SAO FRANCISCO - SE	SAO SEBASTIAO DO RIO PRETO - MG	CERRO BRANCO - RS	SAO DOMINGOS - GO
SANTA ROSA DE LIMA - SE	SENADOR CORTES - MG	CHARRUA - RS	SAO JOAO DA PARAUNA - GO
SAO DOMINGOS - SE	SENHORA DE OLIVEIRA - MG	CHUI - RS	SITIO D'ABADIA - GO
ABAIRA - BA	SENHORA DO PORTO - MG	COQUEIROS DO SUL - RS	URUTAI - GO

APÊNDICE E – Grupos finais atribuídos às capitais e ao distrito federal

Município - UF	Grupo
Aracajú - SE	3
Belém - PA	7
Belo Horizonte - MG	3
Boa Vista - RR	7
Brasília - DF	2
Campo Grande - MS	4
Cuiabá - MT	2
Curitiba - PR	2
Florianópolis - SC	2
Fortaleza - CE	2
Goiânia - GO	2
João Pessoa - PB	2
Macapá - AP	7
Maceió - AL	3
Manaus - AM	7
Natal - RN	3
Palmas - TO	2
Porto Alegre - RS	3
Porto Velho - RO	7
Recife - PE	3
Rio Branco - AC	3
Rio de Janeiro - RJ	2
Salvador - BA	3
São Luís - MA	3
São Paulo - SP	3
Teresina - PI	7
Vitória - ES	2

APÊNDICE F – Gráficos do estudo de simulação sobre escolha de número de clusters

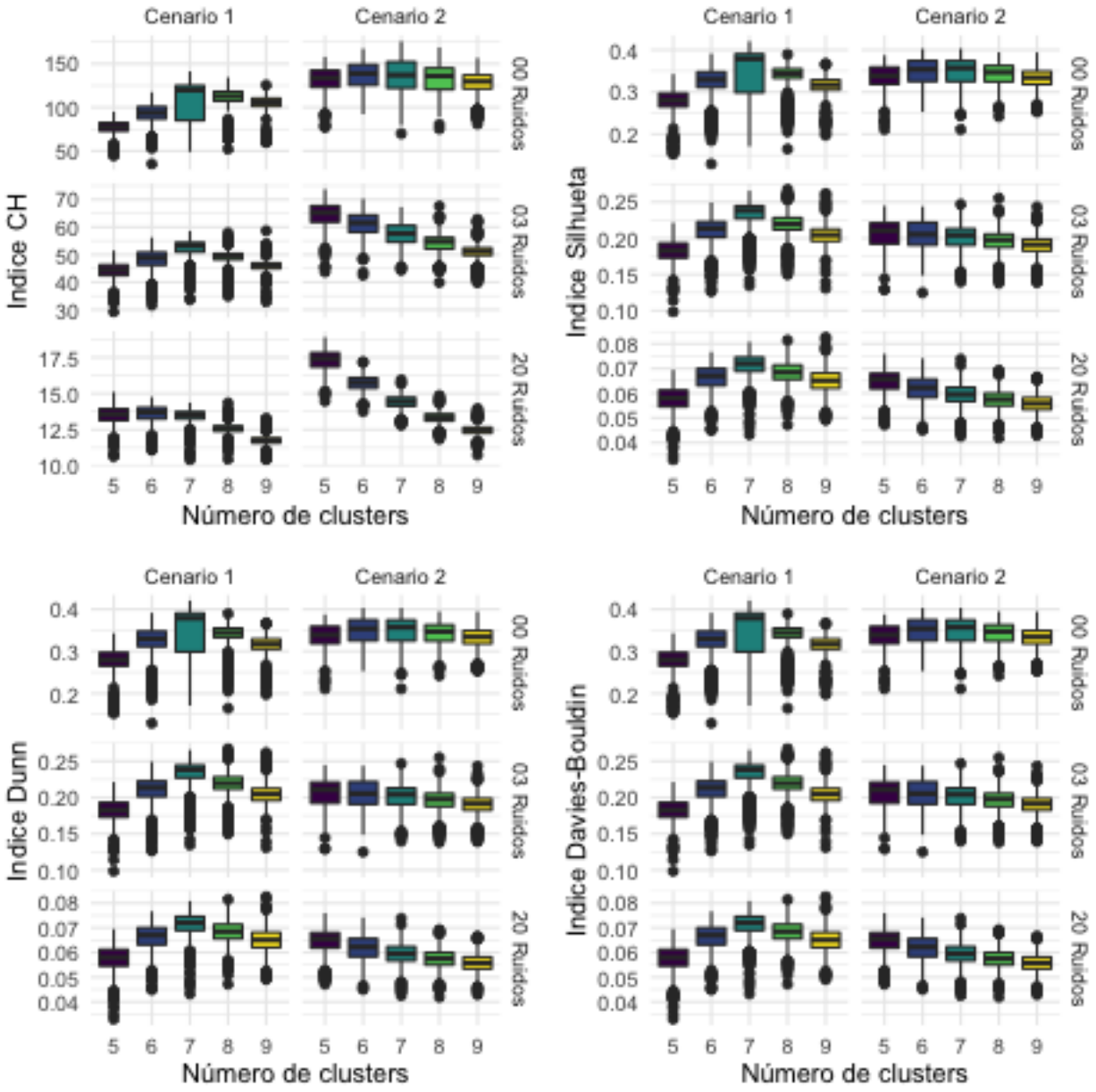


Figura 38 – Gráficos de boxplot do estudo de simulação para os diferentes cenários para comparação da escolha de número de clusters, considerando o método de agrupamento K-médias.

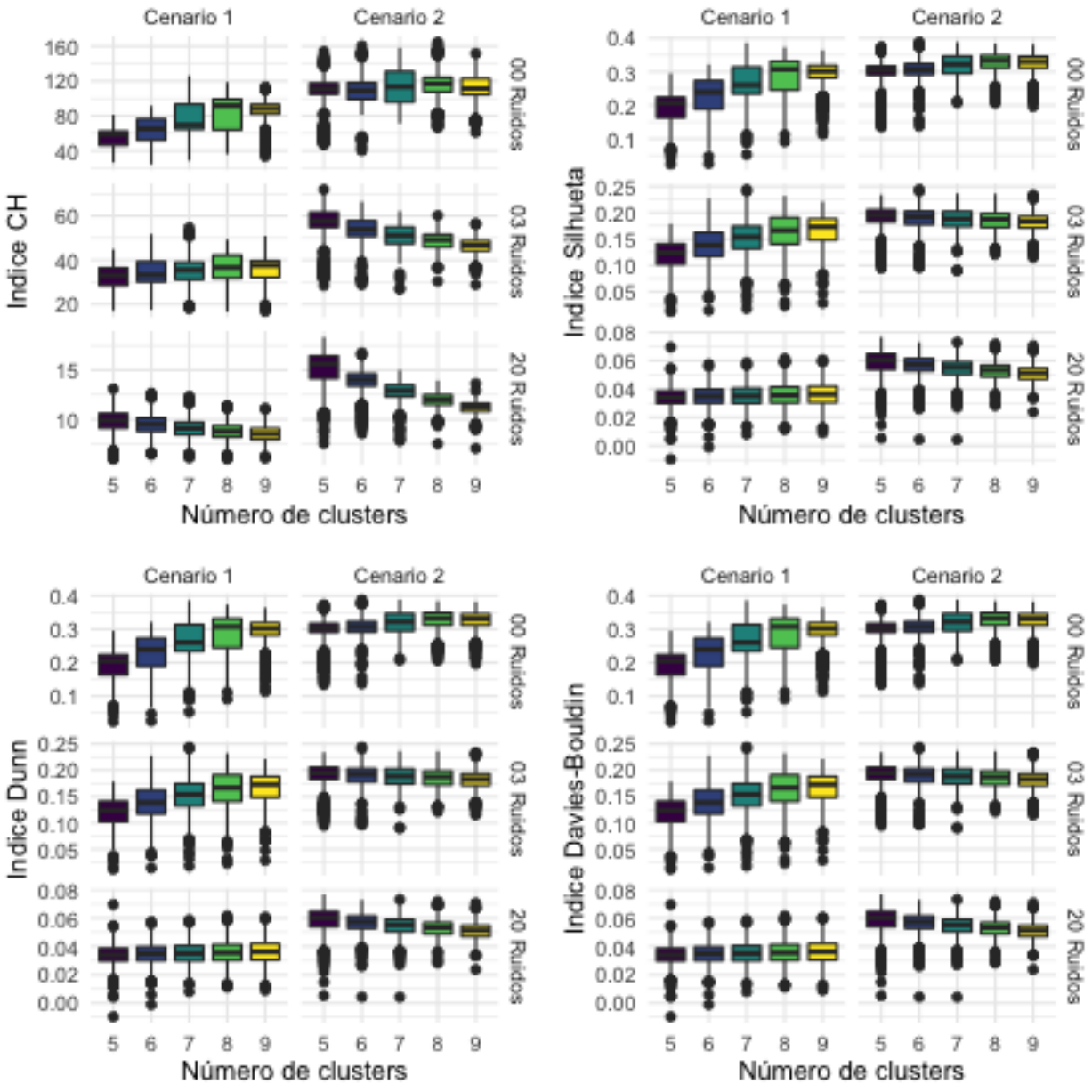


Figura 39 – Gráficos de boxplot do estudo de simulação para os diferentes cenários para comparação da escolha de número de clusters, considerando o método de agrupamento DIANA.

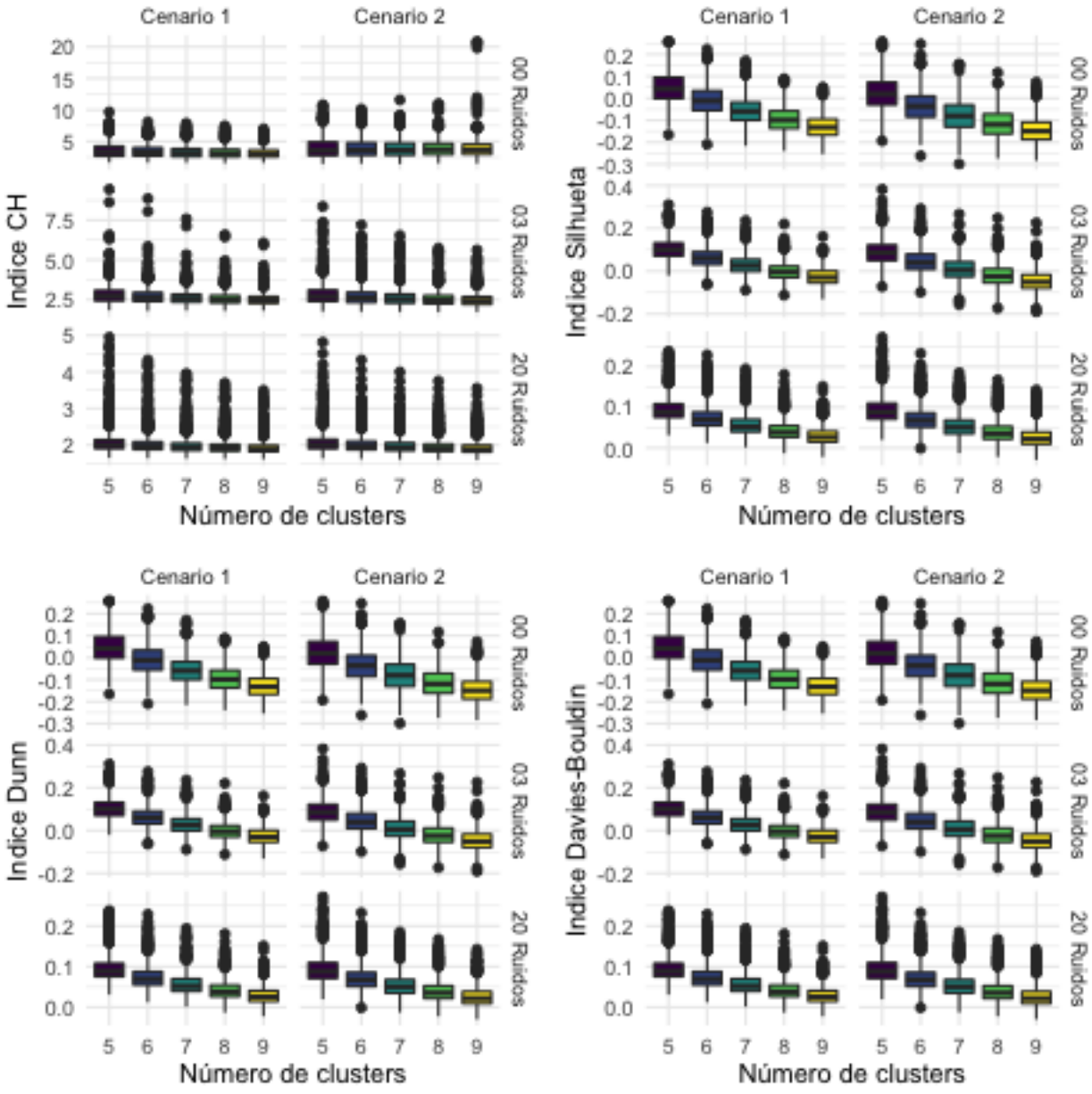


Figura 40 – Gráficos de boxplot do estudo de simulação para os diferentes cenários para comparação da escolha de número de clusters, considerando a técnica de agrupamento hierárquico pelo método do centróide.

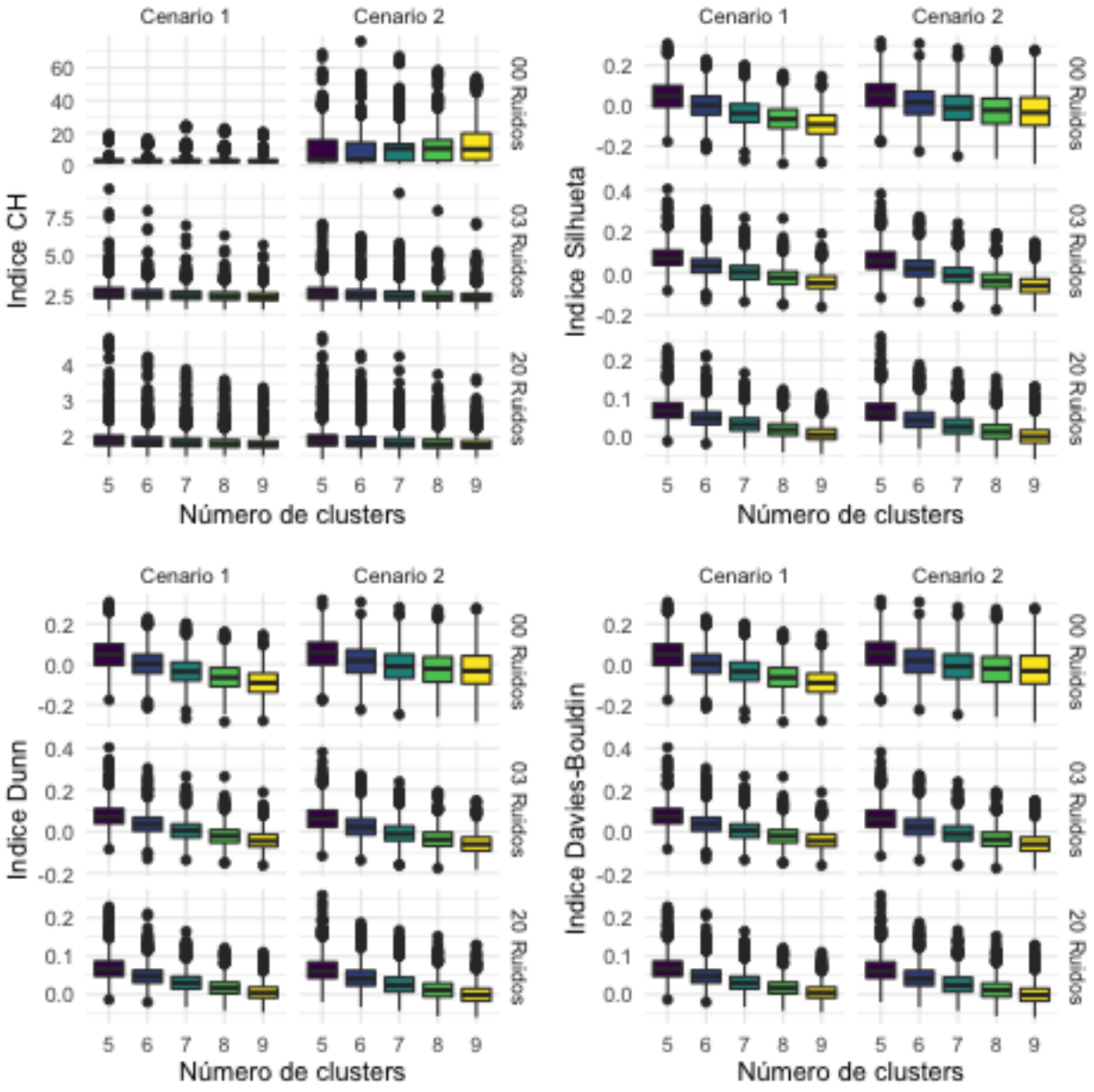


Figura 41 – Gráficos de boxplot do estudo de simulação para os diferentes cenários para comparação da escolha de número de clusters, considerando a técnica de agrupamento hierárquico pelo método do vizinho mais próximo.

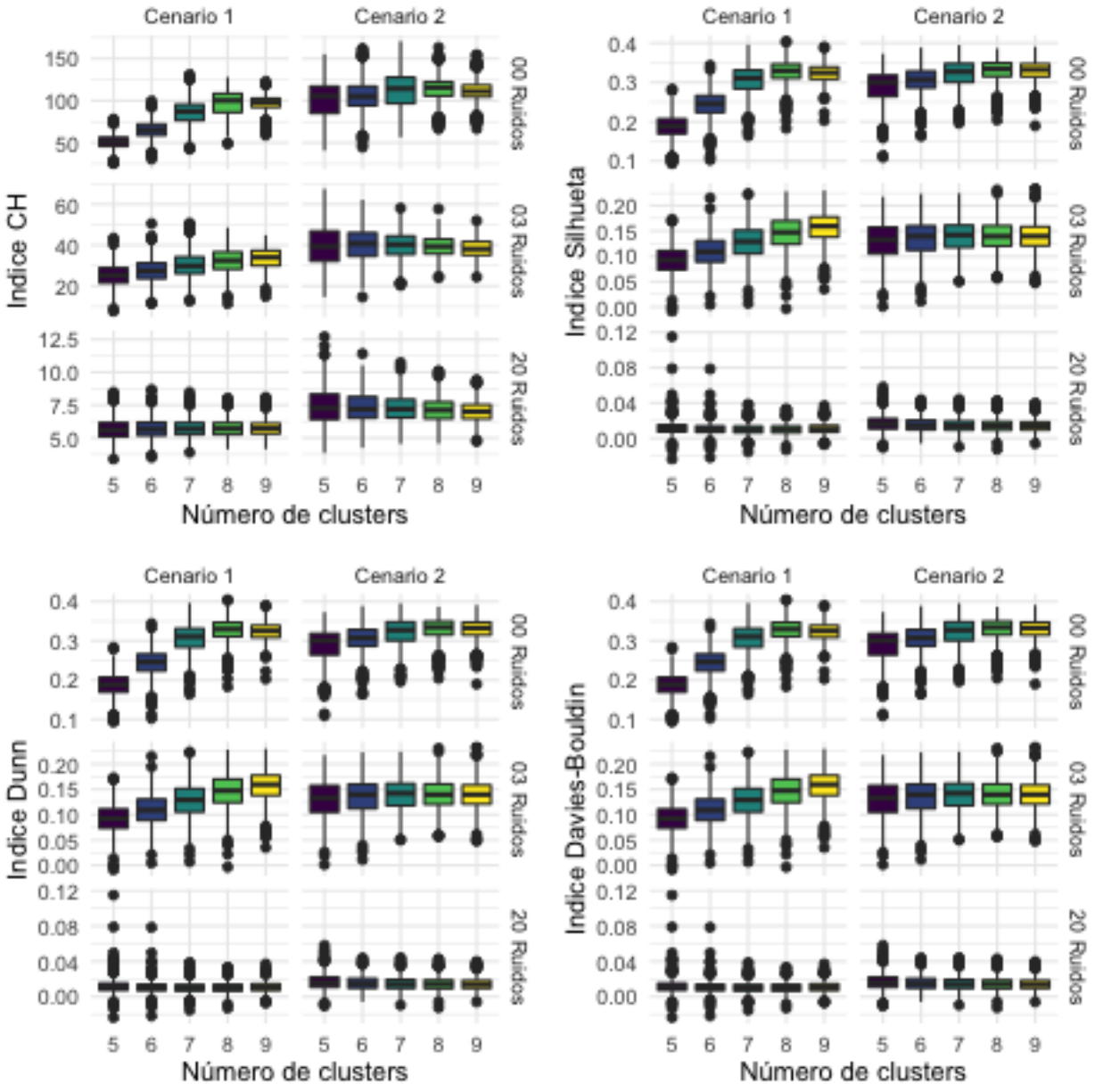


Figura 42 – Gráficos de boxplot do estudo de simulação para os diferentes cenários para comparação da escolha de número de clusters, considerando a técnica de agrupamento hierárquico pelo método do vizinho mais distante.