Universidade Federal da Paraíba
Centro de Informática
Programa de Pós-Graduação em Modelagem Matemática e Computacional

MODELO DE SEGMENTAÇÃO CLUSTERWISE COM PROTÓTIPOS HÍBRIDOS

Wilter da Silva Dias

Orientadores: Prof. Dr. Eufrásio de Andrade Lima Neto
Prof. Dr. Marcelo Rodrigo Portela Ferreira

João Pessoa
Janeiro de 2021

MODELO DE SEGMENTAÇÃO CLUSTERWISE COM PROTÓTIPOS HÍBRIDOS

Wilter da Silva Dias

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO PROGRAMA DE PÓS-GRADUAÇÃO EM MODELAGEM MATEMÁTICA E COMPUTACIONAL (PPGMMC) DO CENTRO DE INFORMÁTICA DA UNIVERSIDADE FEDERAL DA PARAÍBA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM MODELAGEM MATEMÁTICA E COMPUTACIONAL.

Examinada por:

Eufrásio de Andrade Lima Neto
Prof. Dr. Eufrásio de Andrade Lima Neto

Marcelo Rodrigo Portela Ferreira
Prof. Dr. Marcelo Rodrigo Portela Ferreira

Telmo de Menezes e Silva Filho
Prof. Dr. Telmo de Menezes e Silva Filho

Anderson Luiz Ara Souza
Prof. Dr. Anderson Luiz Ara Souza

JOÃO PESSOA, PB – BRASIL
JANEIRO DE 2021

Catalogação na publicação
Seção de Catalogação e Classificação

D541m Dias, Wilter da Silva.
Modelo de segmentação Clusterwise com protótipos híbridos / Wilter da Silva Dias. - João Pessoa, 2021.
120 f. : il.

Orientação: Eufrásio Lima Neto.
Coorientação: Marcelo Ferreira.
Dissertação (Mestrado) - UFPB/CI.

1. Programação de computador. 2. Clusterwise. 3. Regressão. 4. Aprendizagem de Máquina. 5. Alocação. 6. Protótipos Híbridos. I. Lima Neto, Eufrásio. II. Ferreira, Marcelo. III. Título.

UFPB/BC CDU 519.6(043)

Elaborado por ANNA REGINA DA SILVA RIBEIRO - CRB-15/024

"Essencialmente, todos os modelos estão errados, mas alguns são úteis"— George E. P. Box

Agradecimentos

Primeiramente gostaria de agradecer a Deus, devido ao sucesso e a vivência da oportunidade conforme foi dada.

Agradeço aos meus pais Maria (Vera) e Wilson, e ao meu irmão Matheus que sempre estiveram ao meu lado me apoiando ao longo de toda a minha trajetória.

À Laura Diana por se interessar e me dar apoio na minha trajetória até aqui, estando sempre ao meu lado durante o meu percurso acadêmico.

Aos meus orientadores Eufrásio Lima Neto e Marcelo Ferreira por aceitarem e conduzirem o meu trabalho de pesquisa, corrigindo, afastando dúvidas e concedendo orientação em cada etapa. Aos meus examinadores de banca Telmo e Anderson por enriquecerem o presente trabalho com dicas.

Agradeço ao professor José Miguel Aroztegui por me acompanhar e me aceitar no estágio-docência, além de ter sido meu professor duas vezes, na disciplina de Mecânica dos Fluídos na graduação e de Otimização no mestrado, e também pelas conversas e humildade de sua pessoa.

Aos meus colegas Antônio Rubens, Alexandre Gomes, Anderson Morais, Dionarte Dantas, Jefferson Bezerra, Rodrigo Nóbrega, Valdemi Nunes e Gedeão Corpes pelas trocas de ideias, ajuda conjunta e por alguns poucos momentos e conversas que tivemos durante o percurso.

A todos os meus professores do PPGMMC da UFPB pela excelência da qualidade técnica de cada um.

Agradeço à coordenadora do Programa Ana Wyse por ter conseguido bolsas de estudo dentro do seu possível, como também por me ter acompanhado no momento da assinatura dos termos e de ter conduzido o Programa. Também agradeço ao coordenador do Programa nomeado depois, Hugo Cavalcante, por ter me dado suporte e instruído em relação ao Programa.

Ao Gean Paulo e aos outros funcionários da Universidade UFPB que contribuíram de forma direta e indireta para a possibilidade deste trabalho.

À FAPESQ-PB pelo fomento a pesquisa acadêmica subsidiando este trabalho por meio de bolsa acadêmica de mestrado.

Por fim agradeço a todos da UFPB que visam o conhecimento e a propagação deste em benefício de todos.

Resumo da Dissertação apresentada ao PPGMMC/CI/UFPB como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

MODELO DE SEGMENTAÇÃO CLUSTERWISE COM PROTÓTIPOS HÍBRIDOS

Wilter da Silva Dias

Janeiro/2021

Orientadores: Prof. Dr. Eufrásio de Andrade Lima Neto
Prof. Dr. Marcelo Rodrigo Portela Ferreira

Programa: Modelagem Matemática e Computacional

Resumo: Apresenta-se, nesta Dissertação, uma metodologia que combina técnicas de predição e agrupamento denominada Modelo de Segmentação *Clusterwise* com Protótipos Híbridos (MoSCH), o qual objetiva segmentar os dados em *clusters* de modo que cada *cluster* seja representado por um modelo preditivo, como, por exemplo, um modelo de regressão ou algoritmo de aprendizagem de máquina (protótipo), dentre uma lista de métodos pré-definidos. A escolha do melhor protótipo para cada *cluster* tem o intuito de minimizar uma função objetivo. Além da implementação do algoritmo de estimação do método MoSCH, consideramos diferentes técnicas de alocação para novas observações de modo a avaliar o poder preditivo do algoritmo. Uma prova de convergência é apresentada, bem como a aplicação do método proposto em dados sintéticos e a bases de dados reais. Um novo método de alocação baseado no KNN, chamado alocação com KNN dos *clusters* combinados, é proposto, apresentando resultados interessantes. Já no experimento com dados sintéticos o algoritmo MoSCH é comparado com outro algoritmo em 6 cenários diferentes, tendo um desempenho satisfatório. Na validação do algoritmo MoSCH com dados reais, o método proposto apresenta uma relevante performance quando comparado a outros 3 algoritmos (K-means Linear, K-means Híbrido e Regressão Linear Clusterwise), bem como a avaliação de 5 diferentes métodos de alocação.

Palavras-chave: *Clusterwise*, Regressão, Aprendizagem de Máquina, Alocação, Protótipos Híbridos.

Abstract of Dissertation presented to PPGMMC/CI/UFPB as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

CLUSTERWISE SEGMENTATION MODEL WITH HYBRID PROTOTYPES

Wilter da Silva Dias

January/2021

Advisors: Prof. Dr. Eufrásio de Andrade Lima Neto
Prof. Dr. Marcelo Rodrigo Portela Ferreira

Program: Computational Mathematical Modelling

Abstract: This dissertation presents a methodology that combines prediction and grouping techniques called the Clusterwise Segmentation Model with Hybrid Prototypes (CSMoH), which aims to segment the data in clusters so that each cluster is represented by a predictive model, such as a regression model or machine learning algorithm (prototype), among a list of predefined methods. The choice of the best prototype for each cluster is intended to minimize an objective function. In addition to the implementation of the CSMoH method estimation algorithm, we consider different allocation techniques for new observations in order to assess the predictive performance of the algorithm. A proof of convergence is presented, as well as the application of the proposed method in synthetic data and in real databases. A new allocation method based on KNN, called KNN-combining clusters, is proposed, presenting interesting results. In the experiment with synthetic data, the CSMoH algorithm is compared with another algorithm in 6 different scenarios, with an satisfactory performance. In the validation of the CSMoH algorithm with real data, the proposed method presents a relevant performance when compared to 3 other algorithms (Linear K-means, Hybrid K-means and Clusterwise Linear Regression), as well as the evaluation of 5 different allocation methods.
Keywords: Clusterwise, Regression, Machine Learning, Allocation, Hybrid Prototypes.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Um modelo matemático é algo extremamente importante quando se quer entender algo real. Pode ser baseado em conceitos determinísticos como também em conceitos probabilísticos, ou uma combinação dos dois, buscando uma representação da realidade. Um modelo não precisa ser algo extremamente preciso, basta trazer resultados satisfatórios e não necessariamente é preciso abordar todos os contextos possíveis.

Segundo MONARD e BARANAUSKAS [24] a Hierarquia do Aprendizado parte da indução como forma de inferência lógica, e que através dela pode-se ter conclusões genéricas sobre um conjunto particular de dados, formando conceitos por meio de inferências indutiva sobre as observações, sendo que as hipóteses da inferência indutiva podem ou não conservar a verdade. A indução representa um raciocínio que parte de um conceito específico e o generaliza. Para a indução ser viável é necessário uma boa escolha das observações estando essa em uma quantidade suficiente. O sistema de aprendizado indutivo é dividido em supervisionado, em que se conhece o indutor, e o não supervisionado. Sendo que os sistemas de aprendizado podem tanto trazer representações de baixa compreensão como também compreensíveis ao ser humano. Existem diversos paradigmas de aprendizado de máquinas dentre eles o Simbólico, o Estatístico, o baseado em Exemplos, o Conexionista e o Genético.

A aprendizagem de máquina une a estatística com a ciência da computação, tendo termos que estão presentes em ambos, é uma área da Inteligência Artificial (IA) que estuda métodos de aprendizado para a obtenção automática de conhecimento. Se utilizando de um conjunto de dados para um problema projeta-se manualmente as variáveis de entrada e as de saída, e escolhe-se o(s) modelo(s) que potencialmente poderiam descrever o problema, mas para isso é necessário especialização do profissional na seleção dos recursos, habilidades e experiência com *data science*. De acordo com PIERSON [25] são 3 tipos de aprendizagem de máquina: supervisionada, não supervisionada e o semi-supervisionada. Na supervisionada existe uma rotulação dos dados em que sabe-se a natureza, os tipos e as dimensões dos

conjuntos das entradas e das saídas, sendo aplicado em problemas de regressão e de classificação. Na não supervisionada não se sabe bem como os resultados devem aparentar, e não há *feedback* em relação a como os resultados da previsão devem ser, não tendo um "professor"para corrigir, modelos de *clustering* podem identificar perfis/grupos (sem rótulo) os quais não se sabe o que é de fato, não sabendo se a quantidade de perfis/grupos estão certas e se as observações estão agrupadas corretamente. Já a semi-supervisionada situasse entre a aprendizagem de máquina supervisionado e a não supervisionado, é também chamado de aprendizado de reforço.

Este trabalho envolverá temas associados a regressão e análise de agrupamento. O método proposto, chamado de modelo de segmentação *clusterwise* com protótipos híbridos, combina métodos de regressão linear e não-linear, métodos de regressão não paramétricos, algoritmos de aprendizagem de máquina e o método *k-means*, escolhendo o melhor modelo que minimize uma função objetivo. Também investigará a capacidade preditiva do modelo proposto em estimar novas observações.

Na Estatística, a análise clássica de regressão visa estimar uma relação entre uma variável dependente (Y) e um conjunto de variáveis independentes ($X_1, \ldots, X_p$) fazendo uso de uma função que depende de tais variáveis e de um vetor de parâmetros $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)$, que precisam ser estimados. De modo geral, o método dos mínimos quadrados é utilizado para estimar o vetor $\boldsymbol{\beta}$, que consiste em minimizar a soma dos quadrados dos erros e não requer nenhuma suposição probabilística sobre os erros do modelo.

A maneira mais simples de se propor uma relação entre as variáveis é através do modelo de regressão linear. Entretanto, caso a relação entre as variáveis seja não-linear tal modelo se torna inadequado. A obtenção das estimativas para um modelo de regressão não-linear é feita de forma semelhante. Contudo, devido a complexidade de se obter soluções analíticas quando estas existirem, será útil o uso de métodos computacionais iterativos de otimização, o que poderá levar a problemas de convergências destes, mesmo que exista uma solução.

A regressão costuma ser eficiente quando se trabalha com dados homogêneos, ou seja, quando a dispersão dos dados em relação a média é constante. No entanto, quando se fala de dados heterogêneos, modelos com clusterização são uma boa alternativa, pois será possível fazer o uso de métodos de modo a encontrar grupos homogêneos de dados. Na regressão *clusterwise* tem-se um modelo de regressão diferente adotado para cada *cluster* como centroide, combinando assim, análise de agrupamento e regressão. Um *cluster* é formado por um sub-conjunto de observações do conjunto de dados de modo que a dissimilaridade entre esses objetos seja miníma.

Dos trabalhos existentes, relacionados ao tema da dissertação e que fizeram parte

da revisão de literatura, destacam-se:

- CARVALHO *et al.* [4] adaptam a regressão linear *clusterwise* para dados do tipo valor-intervalo, combinando o algoritmo de agrupamento dinâmico com o método de regressão de centro e amplitude para dados do tipo valor-intervalo, abordando vários conceitos sobre o tema. O trabalho de LIMA NETO e CARVALHO [21] foi de grande valia pois apresentam conceitos de regressão não-linear, analisando mais de uma heurística de otimização.

- LOUREIRO [22] faz uso de técnicas de KDD (*Knowledge Discovery in Databases*) para extrair informações úteis em uma extensa quantidade de dados químicos. O trabalho de LOUREIRO [22] foi útil pois serviu como amparo nos conceitos associados as técnicas de análise de agrupamento.

- CARVALHO *et al.* [5] apresentam um modelo de Regressão *Clusterwise* Não-Linear para o Centro e Amplitude em dados do tipo-intervalo (iCRCNLR - *Interval Center and Range Clusterwise Non-Linear Regression*), baseado no algoritmo de agrupamento dinâmico e nos modelos de regressão linear e não-linear para dados tipo-intervalo. O método extrapola o caso linear de regressão *clusterwise*, pois baseia-se em um critério de otimização que seleciona o mais adequado par de modelos (linear e/ou não linear) para centro e amplitude dos intervalos de forma automática. Na questão de alocação para as novas observações definidas em um conjunto teste, foram comparados três métodos: *k-nearest neighbors* (KNN) com distância de Hausdorff, *Stacked Regressions* e alocação aleatória. A presente dissertação pode ser considerada uma continuação do trabalho de CARVALHO *et al.* [5] com certas diferenças, dentre elas, não será usado a abordagem centro-amplitude e por consequência não será trabalhado com dados do tipo-intervalo. Além disso, irá considerar além de modelos lineares e não-lineares paramétricos, modelos não-paramétricos e algoritmos de aprendizagem de máquinas supervisionados.

Esta dissertação está estruturada da seguinte maneira: finaliza-se o Capítulo 1 apresentando os objetivos desta Dissertação; o Capítulo 2 apresenta uma breve revisão sobre regressão linear, análise de agrupamento e alguns conceitos estatísticos envolvidos; o Capítulo 3 apresenta o método de regressão *clusterwise* proposto trazendo um pouco do que foi abordado no Capítulo 2 para a regressão não-linear; o Capítulo 4, tem foco no Modelo de Segmentação *Clusterwise* Híbrido (MoSCH) que configura-se na contribuição desta Dissertação; o Capítulo 5 apresenta uma análise experimental com dados sintéticos para demonstrar a performance do MoSCH; o Capítulo 6, traz aplicações a dados reais para verificar a eficácia da utilização do MoSCH na solução de casos práticos; o Capítulo 7, trará algumas propostas de

trabalhos futuros relacionados aos métodos de alocação; o Capítulo 8 apresenta a conclusão, enfatizando o que foi exposto no trabalho e a relevância dos resultados obtidos; O Apêndice A apresenta de forma detalhada os testes de hipóteses para os métodos de alocação; O Apêndice B apresenta os resultados dos testes de hipóteses dos algoritmos quando comparado os seus melhores resultados encontrados no Capítulo 6.

1.1 Objetivos

- Propor um modelo de segmentação *clusterwise* com protótipos híbridos, considerando uma ampla classe de modelos de regressão e algoritmos de aprendizagem de máquina supervisionados e considerando o método *K-means*.

1.1.1 Objetivos específicos

- Apresentar e implementar um algoritmo de segmentação *clusterwise* com protótipos híbridos baseado no método *K-means*;
- Apresentar a prova de convergência do algoritmo proposto;
- Implementar a alocação de novas observações para o modelo proposto;
- Comparar o modelo proposto com outros existentes na literatura em dados simulados e reais.

Capítulo 2

Regressão e Agrupamento

Neste capítulo serão apresentados assuntos pertinentes a regressão e a análise de agrupamento. Serão explicados na parte de regressão a ideia de estimação, somas dos quadrados e a análise de variância. Já na parte de análise de agrupamento serão apresentadas as medidas de proximidade, a definição de centroide, os métodos de agrupamento por partição de forma detalhada, tendo também uma menção aos métodos hierárquicos e, por fim, será visto algumas medidas de ajustes para *cluster analysis*.

2.1 Regressão

O modelo clássico de regressão teve início nos trabalhos de astronomia propostos por Gauss entre os anos de 1809 e 1821 [11]. Ele é interessante quando se deseja associar um conjunto de variáveis independentes $X_1, X_2, \ldots, X_p$, por meio de uma função matemática, com uma variável resposta Y. Pode-se definir as observações da variável resposta por meio de um vetor $y = (y_1, ..., y_n)^T$, em que cada elemento i representa um elemento amostral, $i = 1, ..., n$ [10]. Supomos, ainda, que cada elemento y_i é independente ou não-correlacionado. Ademais, cada $E(y)_i$ apresenta média μ_i e variância σ^2 constante. A média μ_i é expressa linearmente, em função de um conjunto de variáveis independentes, como $\mu_i = x_i^T\beta$, $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$ sendo um vetor $1 \times n$, e $\beta = (\beta_1, \beta_2, \ldots, \beta_p)$ um vetor $p \times 1$, com p sendo o número de variáveis independentes associado ao elemento de y_i.

Colocando-se na forma matricial tem-se que o vetor de médias $\mu = E(y) = X\beta$, com $y = (y_1, .., y_n)^T$ um vetor $n \times 1$, e X uma matriz $n \times p$ de posto completo em que cada linha é expressa por $x_1^T, ..., x_n^T$. Comumente a hipótese de aditividade entre y e μ é adotada, com $y = \mu + \epsilon$, sendo ϵ um vetor de erros de média zero e variância σ^2 constante. Os erros também são considerados independentes ou não-correlacionados. As variáveis explicativas são aquelas que formam as colunas da matriz modelo X, tendo efeitos aditivos e lineares em relação a resposta y. Geralmente faz-se a primeira

coluna da matriz modelo como 1's, sendo o parâmetro correspondente (β_0) chamado de intercepto[4].

Tendo as variáveis independentes $X_1, X_2, \ldots, X_p$ e a variável resposta Y conhecidas, é possível estimar β a partir do *Método de Mínimos Quadrados* (MMQ), o qual não precisa de nenhuma proposição antecipada sobre a distribuição do vetor y. O método MMQ está relacionado a minimizar uma função objetivo denotada por $\sum_i (y_i - \mu_i)^2$ [17].

2.1.1 Estimação

A notação matricial para regressão clássica se dá por $y = X\beta + \epsilon$, e assim $\mu = E(y) = X\beta$. Desta forma o $X\beta$ representa a linearidade do modelo e o ϵ a parte do efeito aleatório. Admite-se ainda que a matriz de covariância é dada por $Cov(\epsilon) = \sigma^2 I$. A fórmula para a *Soma de Quadrados dos Erros* é dada por $SQE(\beta) = (y - X\beta)^T (y - X\beta)$.

O objetivo é estimar β, para isso é necessário derivar o $SQE(\beta)$ em relação a β para obter um mínimo local de interesse. Utilizando a regra da cadeia ao derivar a formula do $SQE(\beta)$ e igualando a zero encontra-se

$$\frac{\partial SQE(\beta)}{\partial \beta} = X^T(y - X\beta)/2 = 0,$$

assim o *estimador de mínimos quadrados* (EMQ) será denotado por

$$\hat{\beta} = (X^T X)^{-1} X^T y.$$

Uma outra expressão para o EMQ de β em função dos erros não observados pode ser obtida, ao substituir $y = X\beta + \epsilon$ em $\hat{\beta} = (X^T X)^{-1} X^T y$, resultando em $\hat{\beta} = \beta + (X^T X)^{-1} X^T \epsilon$. [29]. Uma peculiaridade é que a diferença entre o $\hat{\beta}$ e o vetor de parâmetros β não pode ser feito através da equação $\hat{\beta} = \beta + (X^T X)^{-1} X^T \epsilon$, pois o erro ϵ não é observado. No entanto, essa expressão é usada para o estudo das propriedades do EMQ $\hat{\beta}$.

Sabe-se que y_i é linear, então quando há uma singularidade o determinante de $X^T X$ é zero, isto é, algumas equações no sistema de equação são dependentes, não tendo a matriz X posto completo, ou seja, o número de equações independentes igual ao posto. Quando se depara com uma singularidade admite-se infinitas soluções. No entanto, quando $X^T X$ é não-singular, admite-se para o sistema de equações que será dado uma única solução para o vetor de parâmetros β [1].

2.1.2 Somas de quadrados

A Soma de Quadrados dos Resíduos (SQR) refere-se ao SQE($\hat{\beta}$) e mede a diferença entre os valores observados do vetor y e o valor estimado pelo modelo $\hat{\mu} = X\hat{\beta}$. Então SQR = SQE($\hat{\beta}$) =$(y - X\hat{\beta})^T(y - X\hat{\beta})$.

A *matriz de projeção* H é dada por

$$H = X(X^TX)^{-1}X^T,$$

em que H é uma matriz idempotente, hermitiana, simétrica, positiva semi-definida. Sendo assim o vetor $\hat{\mu}$, referente aos valores estimados de y, pode ser expressos por

$$\hat{\mu} = Hy = X(X^TX)^{-1}X^Ty,$$

ou seja, representa a projeção ortogonal do vetor das observações y em relação ao espaço construído a partir das colunas da matriz X [11].

O vetor ϵ dos erros não observados pode ser determinado a partir do vetor de resíduos r, em que

$$r = y - \hat{\mu} = y - X\hat{\beta}.$$

Como $\hat{\mu} = X\hat{\beta} = Hy$, então

$$r = y - Hy = (I - H)y,$$

sendo I a matriz identidade de ordem n.

2.1.3 Análise de variância

Para verificar se o modelo de regressão linear é adequado é possível utilizar a técnica de análise de variância, baseada na seguinte relação:

$$\sum_i (y_i - \bar{y})^2 = \sum_i (\hat{\mu}_i - \bar{y})^2 + \sum_i (y_i - \hat{\mu}_i)^2$$

Assim, a soma dos quadrados das observações em relação ao valor médio $\bar{y}$ (SQT) é igual a soma dos quadrados explicadas pelo modelo de regressão (SQE) mais a soma dos quadrados residual (SQR), esta não-explicada pelo modelo. Tanto o SQE como o SQR possuem distribuições independentes $\sigma^2 X^2_{(p-1)}$ e $\sigma^2 X^2_{(n-p)}$. O coeficiente de correlação múltipla de Pearson (R^2) é detonado por

$$R^2 = SQE/SQT$$

e assume valor entre 0 e 1. Um bom modelo de regressão linear deve apresentar

elevado R^2, bem como valores pequenos para a estimativa σ^2, visto que os intervalos de confianças de interesse são proporcionais ao $\hat{\sigma}$.[11]

2.2 Análise de agrupamento

A análise de agrupamento é uma importante tarefa para descobrir estruturas em conjuntos de dados e vem sendo amplamente empregado em diversas áreas do conhecimento, tais como, mineração de dados, recuperação de informação, sistemas de recomendação, taxonomia, dentre outras. Ela tem por objetivo agrupar objetos a partir da maximização da similaridade nos grupos/classes ou agrupar conforme a minimização da dissimilaridade entre grupos/classes, ou seja, maximização da similaridade dentro das classes e minimização da similaridade entre as classes.

O método de agrupamento pode ser particional (divide o conjunto de dados em grupos não sobrepostos de modo que cada objeto está em somente um grupo) ou hierárquico (constrói uma hierarquia de partições no conjunto de dados, de forma aglomerativa ou divisiva). Existem dois paradigmas de agrupamento particional, o exclusivo (*hard*) em que cada objeto deve pertencer exclusivamente a um único grupo e o não exclusivo (fuzzy) em que um objeto pertence a todos os grupos com determinado grau de pertinência para cada grupo [3].

A partir de uma quantidade finita de objetos ou indivíduos, os quais são descritos por determinados atributos, isto é, $\Omega = X = \{X_1, ..., X_n\}$ e para cada X_i com $i \leq n$ existe um vetor de propriedades associado $X_i = (x_{i1}, ..., x_{ip}) \in \Re^p$ com $n, i, p \in \aleph^*$ sendo as entradas primárias para o agrupamento.

O objetivo do problema, no caso do *hard*, é encontrar grupos/classes de objetos ou de indivíduos de modo que $\Omega = \Omega_1 \bigcup \Omega_2 \bigcup ... \bigcup \Omega_{q-1} \bigcup \Omega_q$, e que $\Omega_j \bigcap \Omega_k = \emptyset$ com $j, k \in \aleph^*$ sendo $j, k \leq q$ e $j \neq k$, assim $\Omega_j \subseteq \Omega$. Para isso é necessário satisfazer certas condições:

I) a similaridade dentro dos grupos é máxima, e

II) a similaridade entre os grupos é mínima.

A primeira condição se atenta em saber o quão parecido um individuo é do outro para pertence a um determinado grupo, ou seja, tende a resguardar determinadas propriedades de coesão em relação aos indivíduos. A segunda condição está associada a quão longínquo um indivíduo pode estar do outro em relação a semelhança, de modo até mesmo a não poder estar dentro de um mesmo grupo.

Existem diversos algoritmos e métodos que tentam solucionar o problema geral de agrupamento, produzindo uma classificação entre os indivíduos, sendo que pode gerar algo relevante ou não para o problema. É necessário tomar diversas decisões ao escolher um algoritmo ou método, como filtragem de dados, que medida de similaridade usar, como os grupos serão feitos, entre outros fatores. Muitas vezes a

experiência do desenvolvedor em relação ao domínio e das técnicas faz uma vasta diferença.

Um maior aprofundamento sobre a análise de agrupamento pode ser obtido em LOUREIRO [22], RIPLEY [30], R. O. DUDA [26], KAUFMAN L. [18] e R. O. DUDA [27].

2.2.1 Medidas de proximidade

Partindo do pressuposto de que quanto mais um indivíduo/objeto está próximo de um outro em relação aos seus atributos, mais parecidos eles são, ou caso contrário são mais diferentes, então é necessário uma medida de distância partindo da ideia de similaridade ou dissimilaridade. A escolha de uma medida pode estar atrelada também a natureza das variáveis, podendo ser inteira/discreta, real/flutuante/contínua, como também em relação a uma escala como qualitativas, intervaladas, proporcionais, nominais, ordinais e também na experiência, no conhecimento e no embasamento teórico.

Muitas vezes uma matriz de dimensão $n \times n$ é usada para por as relações entre os objetos/indivíduos para informar uma determinada medida de similaridade ou o caso contrário, dissimilaridade. Para isso há a notação que indica o valor da similaridade s_{ij} ou da dissimilaridade d_{ij}, em que i refere-se a um objeto/indivíduo e j a outro objeto/indivíduo, sendo $i < n$ e $j < n$ e $i, j, n \in \aleph^*$. Como na maioria dos casos a matriz é simétrica, e o valor de similaridade ou dissimilaridade de um objeto para ele mesmo é sempre fixo com valor absoluto máximo ou valor nulo, ignora-se a diagonal da matriz e utiliza-se a matriz como uma matriz triangular.

As seguintes condições devem ser satisfeitas entre os objetos i e j:

I)A similaridade ou dissimilaridade deve ser não negativa, $s_{ij} \geq 0$ ou $d_{ij} \geq 0$

II)A similaridade é absoluta máxima ou a dissimilaridade é nula quando vista sobre o mesmo elementos/objetos/indivíduos, s_{ii} =valor absoluto máximo ou $d_{ii} = 0$

III)A similaridade ou dissimilaridade é simétrica para os objetos/indivíduos i e j ou j e i, assim $s_{ij} = s_{ji}$ ou $d_{ij} = d_{ji}$

Para ser uma métrica a matriz de dissimilaridade deve satisfazer a seguinte condição de desigualdade triangular $d_{ij} \leq d_{it} + d_{tj}$, e no caso da similaridade $s_{ij} \geq s_{it} + s_{tj}$, em que $i, j, k \in \aleph^*$ com $i < n$, $j < n$ e $t < n$ sendo os objetos/indivíduos.

Neste trabalho serão usadas medidas de dissimilaridade, devido ao fato de que não é preciso conhecer o maior valor de distância ou não ter problemas com indeterminações. Dentre as medidas mais utilizadas estão as mais simples como as baseadas na métrica de Minkowski, como a métrica euclidiana $r = 2$, a métrica de Manhattan $r = 1$, e a métrica máxima denominada de métrica de Chebyshev (quando r tende

ao infinito), dadas por:

$$d_{ij} = (\sum_{k=1}^{p} |x_{ik} - x_{jk}|^r)^{1/r},$$

em que x_{ik} e x_{jk} são os valores do atributo/propriedade k para o objeto/indivíduo i e para o objeto/indivíduo j, sendo que cada objeto/indivíduo tem p atributos/propriedades. A métrica de Minkowski obtém uma distância de ordem r a partir da diferença entre os atributos/propriedades $X_i = \{x_{i1}, x_{i2}, ..., x_{ip}\}$ e $X_j = \{x_{j1}, x_{j2}, ..., x_{jp}\}$. Outro exemplo é a distância de Canberra, dada por

$$d_{ij} = \begin{cases} 0 & \text{se } x_{ik} = x_{jk} = 0 \\ (\sum_{k=1}^{p} |x_{ik} - x_{jk}|^r)^{1/r} & \text{caso contrário.} \end{cases}$$

Mais detalhes sobre medidas de dissimilaridades, medidas multivariadas contínuas, binárias, ordinais, nominais ou mistas podem ser vistas em GOWER *et al.* [13], GOWER *et al.* [14], GOWER e WARRENS [15], LANCE e WILLIAMS [20], EVERITT *et al.* [9], KAUFMAN L. [18], CUNHA [8] e JOHNSON [16].

2.2.2 centroide

O centroide é uma representação central de um conjunto de dados podendo ser, por exemplo, o objeto/indivíduo que está mais ao centro, uma média, uma mediana, uma moda de uma distribuição, um centro geométrico ou um centro gravitacional (centro de massa). Há também o conceito de pseudo-centroide em que se pode utilizar algoritmos como MinMax-*centroids* e (*weighted*) MinSum-*centroids* [12]. Quando se refere a *cluster*, o centroide é o ponto mais representativo em relação ao grupo. Ele representa todos os objetos/indivíduos do grupo, podendo ser ou não parte do conjunto de dados. No entanto, geralmente se utiliza a média entre os objetos/indivíduos como centroide.

A importância de um centroide está na similaridade deste com os membros do *cluster*, sendo também usado para medir a dissimilaridade entre *clusters*, de tal forma que quanto maior for essa dissimilaridade, melhor será a clusterização. Além disso, permite também poder avaliar o quão dispersos os objetos/indivíduos estão dentro do *cluster*. Outro aspecto importante está em interpretar qualitativamente o tipo de grupo que cada *cluster* representa [6].

Muitas vezes a escolha inicial dos centroides é algo crucial e tem influência no resultado final, devido ao fato dos centroides serem re-computados a cada passo, de modo a otimizar uma função objetivo [34]. Muitas vezes os centroides são escolhidos randomicamente fazendo com que a convergência possa ser mais rápida e satisfatória (quase sempre não ótima).

2.2.3 Métodos de agrupamento por partição

O objetivo dos métodos de partição é obter uma partição única dos objetos em um número fixo de grupos g, tipicamente através da otimização (geralmente local) de uma função objetivo apropriada, que mede a adequação entre grupos e centroides.

Iniciando com g grupos selecionados aleatoriamente, o método por partição faz uso de estratégias para minimizar ou maximizar uma função objetivo de forma iterativa. A função objetivo muda de acordo com a partição que está sendo usada do conjunto de dados. A definição da função objetivo é dada por:

$$F : P_g(\Omega) \rightarrow \Re$$

Sendo que $P_g(\Omega)$ remete ao conjunto de dados para todas as partições. Com a partição do conjunto de dados $\{\Omega_1, \Omega_2, ..., \Omega_g\}$, e cada $\Omega = \{\omega_1, \omega_2, ..., \omega_n\}$, tendo g grupos diferentes de vazio, e com ω_i sendo um vetor p-dimensional, com $i \leq n$ e $i \in \aleph^*$.

Método K-Means

Dependendo da escolha dos centroides iniciais, o *K-Means* tem grande eficiência em relação aos algoritmos hierárquicos tradicionais, Seção 2.2.4. O *K-Means* é um algoritmo que utiliza centroides como pontos que servem para representar grupos/partições, ele foi inicialmente exposto em 1967 [23]. A solução do algoritmo é um ótimo local, assim como a maioria dos métodos que envolvem otimização, e a solução depende das escolhas dos centroides iniciais. Existem técnicas para as escolhas destes centroides, como escolher em uma determinada ordem ou selecionar vários iniciais. Existem também desvantagens em relação ao método como:

- Assumir a quantidade g como conhecida, o que nem sempre é possível definir em aplicações reais.
- As condições iniciais é um fator crucial para a solução, como os grupos inicias e a ordem os objetos.
- Há limitações na convergência para um mínimo local.

O algoritmo do *K-Means* é apresentado na Figura 2.1:

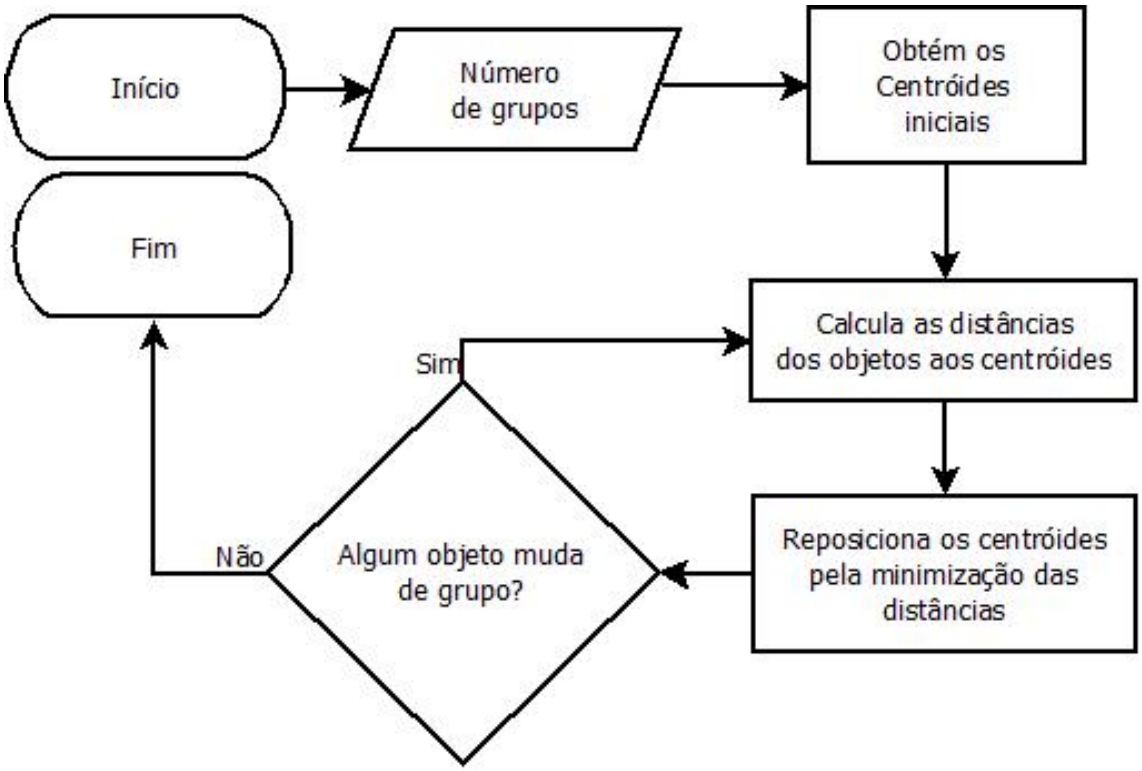

Figura 2.1: Fluxograma do algoritmo K-Means

Autoria própria .

Método PAM

O método PAM (*Partition Around Medoids*) foi proposto em 1987[19]. Tem como objetivo encontrar g grupos, sobre uma quantidade n de indivíduos/objetos, a partir dos medoides (objetos/indivíduos centrais) que são os g objetos/indivíduos representantes de cada grupos (protótipos), sendo $g \leq n$.

O medoides são calculados de modo que a dissimilaridade seja mínima para o centro mais próximo, tendo em vista um subconjunto tal que $m_1, ..., m_k \subset \omega_1, \ldots, \omega_n$ de modo a minimizar a função objetivo:

$$\sum_{i=1}^{n} min\ d(\omega_i, m_k), k = 1, ..., g$$

Cada objeto/indivíduo ω_i é associado ao centroide mais perto o qual é relacionado ao grupo Ω_g, com $i \leq n$ e $i \in \aleph^*$, se o centroide m_{Ω_g} é o mais próximo entre os m_k em relação a ω_i então este será associado a tal grupo de centroide m_{Ω_g}.

$$d(\omega_i, m_{\omega_g}) \leq d(\omega_i, m_k) \forall k = 1, ..., g$$

A cada momento o algoritmo PAM tenta descobrir os centros mais favoráveis. É analisado par a par de objetos/indivíduos de modo que um dos objetos no par é o centro, os pares são formados pelas permutações dos objetos dois a dois.

Uma desvantagem desse algoritmo é que ele é usado apenas para conjuntos de valores g e n pequenos, devido ao alto custo computacional. No entanto ele é bem mais satisfatório que o *K-Means* devido a pouca sensibilidade aos *outliers*. O algorítmo é dado da seguinte forma na Figura 2.2:

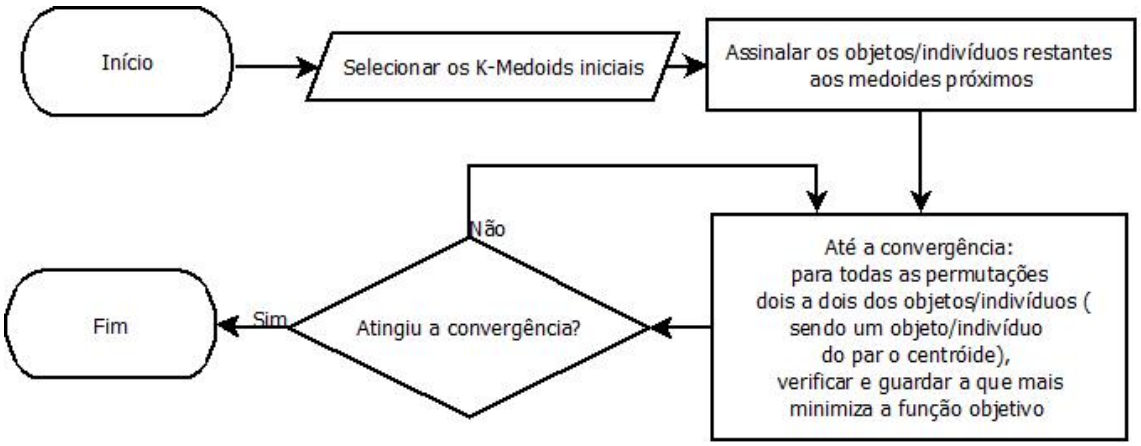

Figura 2.2: Fluxograma do algoritmo PAM

Autoria própria.

Método CLARA

O método CLARA [18], ao contrário do método PAM, não armazena toda a matriz de dissimilaridade, visto que armazenar e processar matrizes muito grandes tende a ser um problema computacional. Existem dois passos essenciais para serem executados no método, no primeiro obtém-se uma amostra dos dados e então agrupa-se em g subconjuntos usando o método PAM que por sua vez fornecerá g objetos/indivíduos representativos. No segundo passo, cada objeto/indivíduo que não pertencer a uma amostra deve ser associado ao objeto/indivíduo mais perto.

É necessário uma matriz de medidas e não uma matriz de dissimilaridade para o método, a qual possui uma dimensão $n \times p$. O método CLARA possui um tempo linear $O(n)$ ao invés de $O(n^2)$ como ocorre no método PAM. A Figura 2.3 mostra como funciona o método.

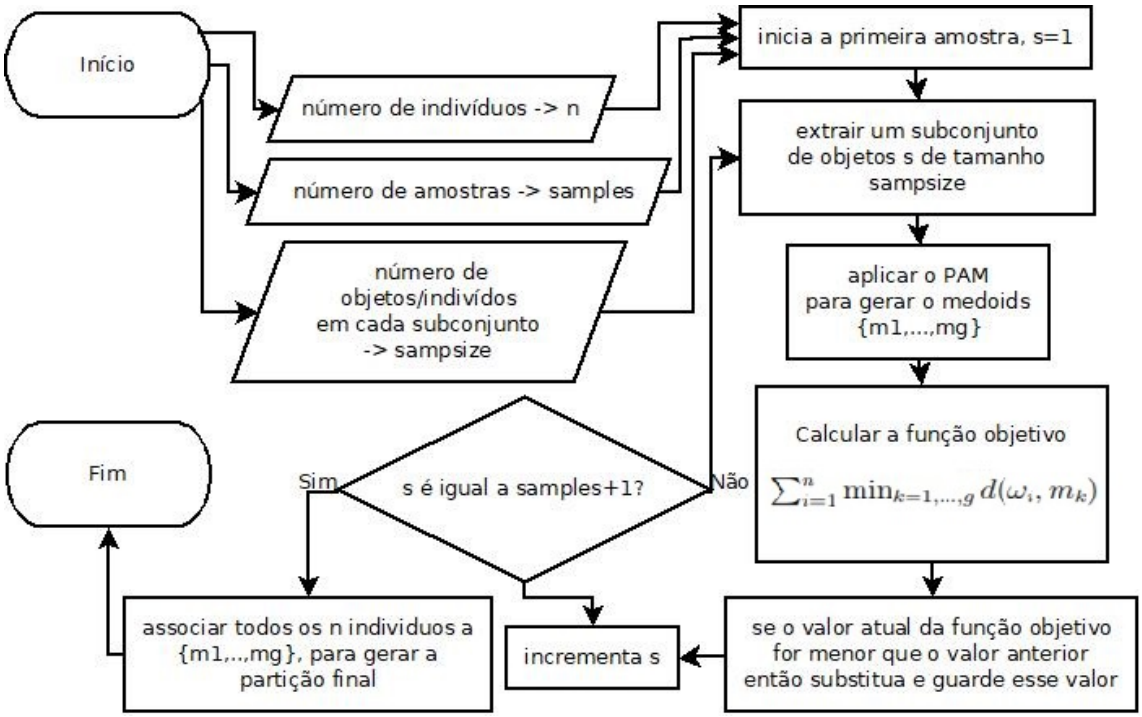

Figura 2.3: Fluxograma do algoritmo CLARA

Baseado na dissertação de LOUREIRO [22].

Método Fuzzy K-means

O método Fuzzy propõe uma certa difusão entre os *clusters*, diferentemente dos métodos já mencionados em que seleciona-se os objetos/indivíduos para um determinado *cluster*. O método permite que certos objetos/indivíduos possam estar relacionados a um ou mais grupos com um certo grau de proximidade. Sendo assim, neste método se trabalha com conjuntos "nebulosos", em que um certo objeto pode, por exemplo, pertencer com probabilidade de 50% a um *cluster* A, 30% a um *cluster* B e 20% a um *cluster* C, permitindo ambiguidades nos dados, que por objetivo costumam ocorrer. Cada objeto/indivíduo ω_i tem um valor u_{ik} associado que representa o quão esse objeto pertence ao grupo Ω_k.

É necessário satisfazer as seguintes condições:

- $u_{ik} \geq 0$ para todo $i = 1, ..., n$ e $k = 1, ..., k$;
- $\sum_{k=1}^{g} u_{ik} = 1 = 100\%$ para todo $i = 1, ..., n$.

O método de Fuzzy foi inicialmente proposto em 1990 [18]. A função $d(i, j)$ retorna a medida de dissimilaridade. No entanto, a medida u_{ik} deve ser conhecida através da minimização da função objetivo:

$$\sum_{k=1}^{g} \frac{\sum_{ij=1} u_{ik}^2 u_{jk}^2 d(i,j)}{2 \sum_{j=n}^{n} u_{jk}^2}$$

A resolução dessa função é da forma não-linear, sendo necessário métodos numéricos para encontrar uma solução local viável, tendo em vista também as restrições nos elementos por meio dos multiplicadores de Lagrange.

2.2.4 Métodos de agrupamento hierárquico

O objetivo de um algoritmo hierárquico é que, dado uma entrada com as possíveis variações $g = \{1, .., n\}$, seja possível construir a partição $P = \{P_1, ..., P_n\}$. Assim, uma partição formada pelo grupo $g = 1$, significa que todos os objetos/indivíduos estão nesse grupo, e o caso contrário, se há n grupos, $g = n$, então os n objetos/indivíduos estarão cada um em um grupo, ou seja, um grupo para cada objeto/indivíduo.

Existem dois tipos de técnicas utilizadas nos métodos hierárquicos, a aglomerativa, a qual agrupam-se *clusters* (inicialmente formados por um objeto/indivíduo) a cada iteração, até o momento em que terá um *cluster* formado por todos os objetos/indivíduos; e a divisiva a qual divide-se um *cluster* (inicialmente formado por todos os objetos/indivíduos) em dois a cada iteração, até o momento em que não se pode mais dividir pois todos os objetos/indivíduos formam um *cluster*. Em nenhuma das duas técnicas há a possibilidade de retroceder, ou seja, uma vez que um *cluster* foi juntado no método aglomerativo ou dividido no método divisivo, os objetos não poderão pertencer a outros clusters em passos posteriores. [22]

Neste trabalho não utiliza método hierárquico. No entanto, pode-se destacar o Método *AGlomerative Nesting* (AGNES) e Método *DIvisive ANAlysis* (Diana).

2.2.5 Medidas de ajuste em análise de agrupamento

É comum utilizar certas ferramentas que trazem algum tipo de análise, podendo elas serem gráficas ou apenas numéricas. As ferramentas gráficas tendem a mostrar uma certa relação no conjunto de dados de forma visual que muitas vezes não é possível entender numericamente de forma manual e concisa.

Dendrograma

Dendro é um termo que exprime a ideia de árvore, ou seja, um dendrograma é um diagrama em forma de árvore que exibe os *clusters*/grupos formados no agrupamento, sendo usado para visualizar como os grupos são unidos ou separados a cada passo do algoritmo de agrupamento, sendo o nível de similaridade ou dissimilaridade medido no eixo vertical e as observações no eixo horizontal. A Figura 2.4 mostra um exemplo de dendograma.

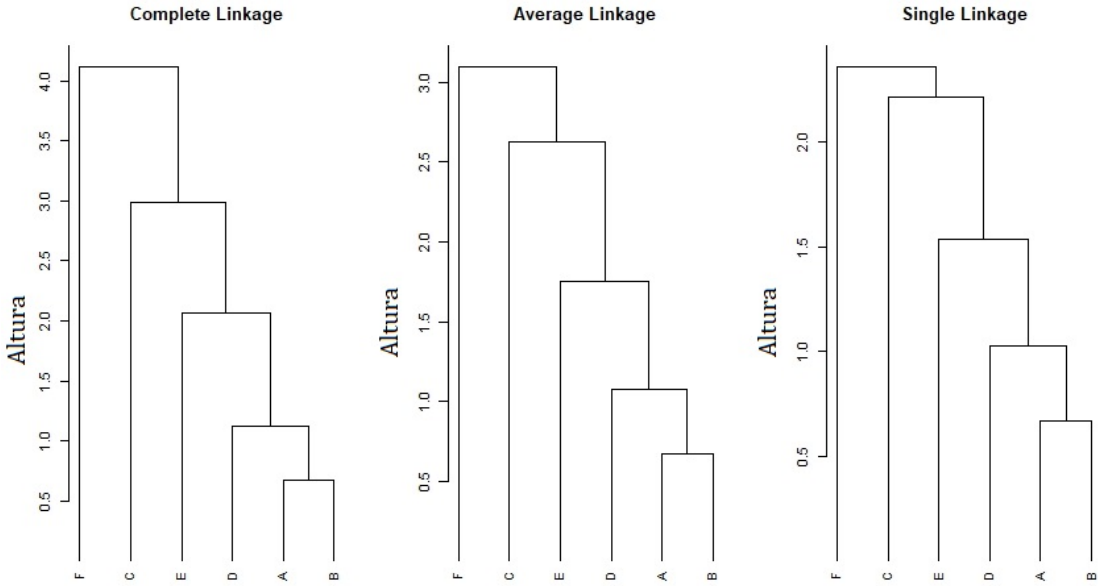

Figura 2.4: Exemplo de dendrograma.

Autoria própria.

Gráfico de silhueta

O gráfico de silhueta é um método para interpretação e validação da consistência de uma clusterização. O gráfico da uma ideia de coesão, pois informa o quão similar um objeto está em relação ao próprio *cluster* e também de separação, pois também é possível comparar a outros *clusters*/grupos. A silhueta varia entre -1 e +1: um valor alto positivo seria conveniente pois quanto mais objetos tiverem valores positivos e altos mais bem combinados eles foram, e o caso contrário em relação aos valores negativos. A silhueta pode usar alguma medida de norma, como a distância euclidiana ou a distância de Manhattan[2, 18, 32, 33]. A regra é elaborada da seguinte forma:

$$s_i = \begin{cases} 1 - a_i/b_i & \text{se } a_i < b_i \\ 0 & \text{se } a_i = b_i \\ b_i/a_i - 1 & \text{se } a_i > b_i \end{cases}$$

em que a_i é a distância média entre o objeto i e todos os outros objetos dentro do mesmo *cluster*/grupo, e b_i a distância média entre o objeto i e todos os outros objetos dos outros *clusters*/grupos. A Figura 2.5 mostra um exemplo de gráfico de silhueta.

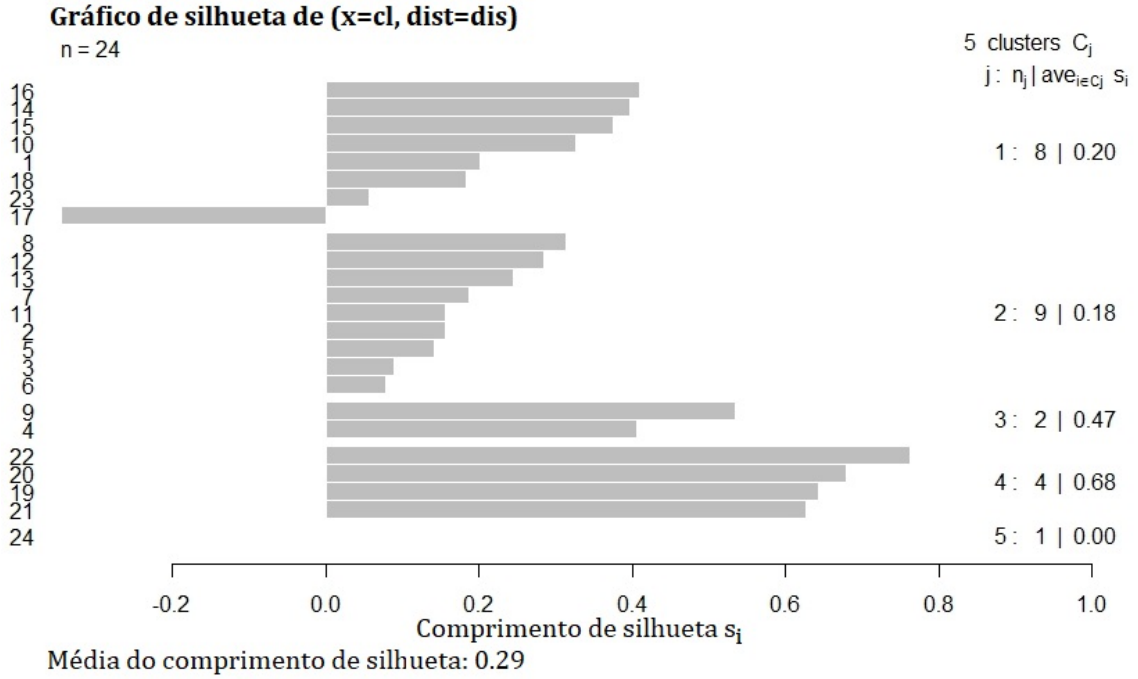

Figura 2.5: Exemplo de gráfico de silhueta

Autoria própria.

Diagrama de dispersão

Um diagrama de dispersão é uma representação gráfica de dados com duas variáveis. É comumente utilizado para verificar se há uma relação entre duas variáveis quantitativas, ilustrando a relação entre elas, podendo ser observado tendências importantes em relação aos dados. Há também uma ideia de correlação negativa entre um par de variáveis quando elas são inversamente proporcionais ou uma ideia de correlação positiva quando são diretamente proporcionais. O gráfico de dispersão é exemplificado na Figura 2.6.

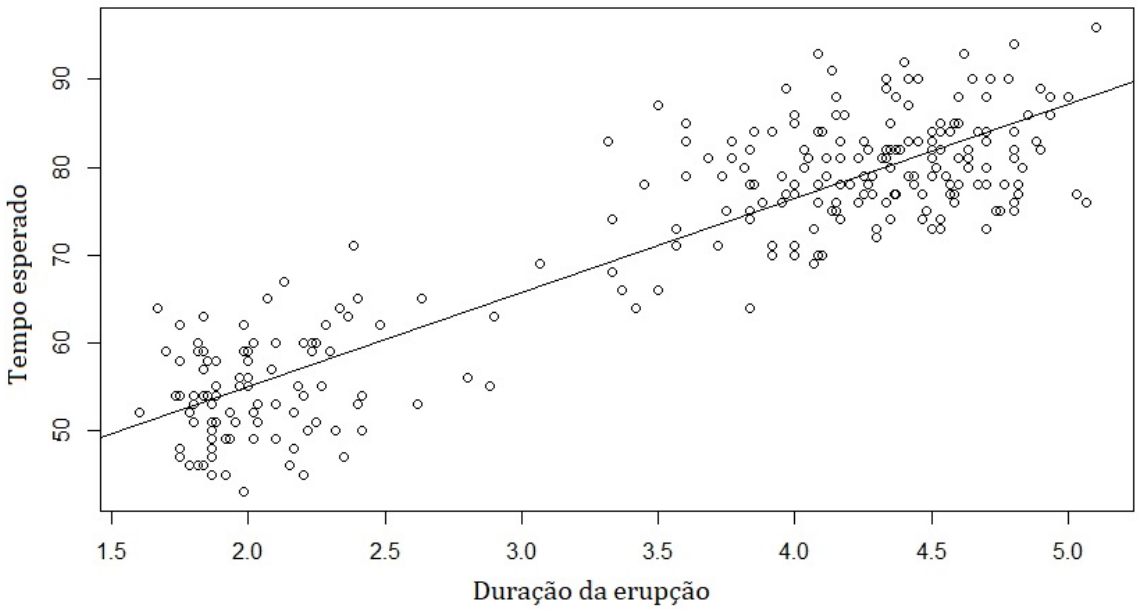

Figura 2.6: Exemplo de gráfico de dispersão

Autoria própria.

Método Elbow

O método Elbow é um método de interpretação e validação da consistência da clusterização e serve para ajudar a encontrar a quantidade apropriada de *clusters* para um conjunto de dados. O método avalia a porcentagem da variância explicada em função do número de *clusters*. O número de *clusters* é escolhido no "ponto de cotovelo", apesar dele não ser tão fácil de ser identificado, mas é o ponto em que o ganho marginal decai de modo a formar uma angulação no gráfico, devido ao fato das primeiras quantidades de *clusters* trazer muita informação. Um exemplo do gráfico de Elbow é visto na Figura 2.7. Podem ser verificados outros critérios, no lugar da variância explicada, como o erro total, a média de dispersão, a soma de quadrados, entre outros.

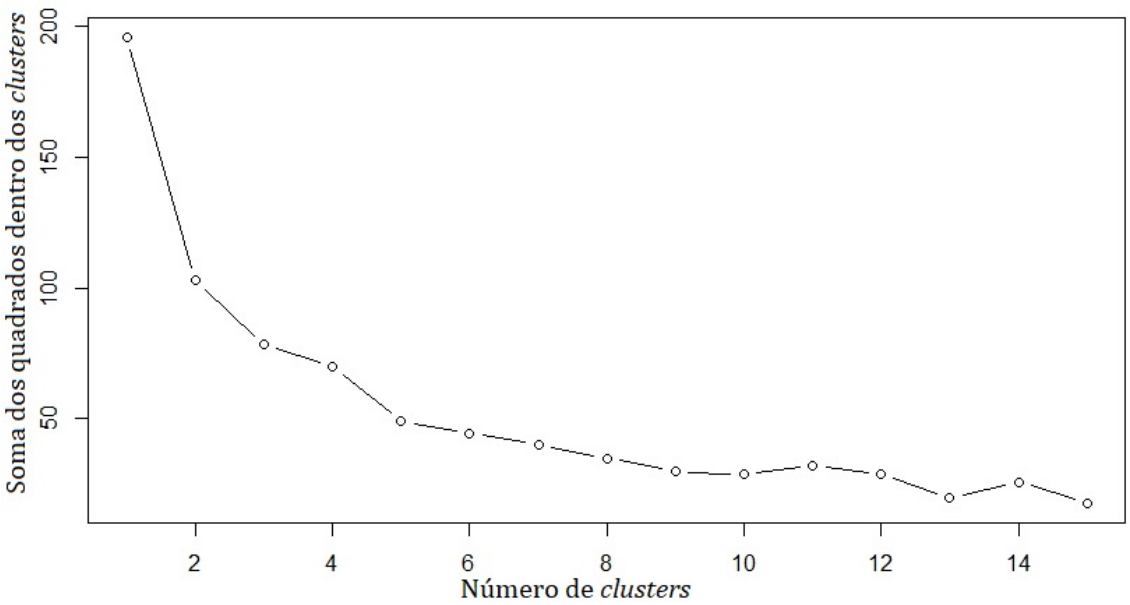

Figura 2.7: Exemplo de gráfico de Elbow. Um possível cotovelo é a quantidade de *clusters* 6

Autoria própria.

Dunn Index

O Dunn Index é um exemplo de combinação não linear de compacidade e separação. Ele é a razão da menor distância entre os objetos/indivíduos que não estão no mesmo *cluster* e à maior distância dos objetos que estão no mesmo *cluster*. A função $dis(i, j)$ calcula a distância entre dois objetos/indivíduos, enquanto que a função *diam* o tamanho do *cluster*. [2][32][33]

$$D(\Omega) = \frac{min_{\omega_k,\omega_l \in \Omega, \omega_k \neq \omega_l} \ (min_{i \in \omega_k, j \in \omega_l} \ dis(i,j))}{max_{\omega_m \in \Omega} \ diam(\omega_m)}$$

Conectividade

Na fórmula da conectividade o termo nomeado como N representa a quantidade de objetos/indivíduos no conjunto de dados (linhas), o M (o qual não é mostrado) é a quantidade de propriedades/atributos dos objetos/indivíduos no conjunto de dados (colunas), e o L é quantidade de vizinhos próximos que serão usados. O termo $t_{i(j)}$ é o j-ésimo vizinho mais próximo do objeto/indivíduo i, no entanto se o objeto i e o objeto j pertencerem ao mesmo *cluster* então $x_{i,t_{i(j)}} = 0$ e caso contrário $x_{i,t_{i(j)}} = 1/j$. As partições devem ser definidas como $\Omega = \{\omega_1, ..., \omega_g\}$ para os N objetos dentro dos g *clusters*. A conectividade poderá ter valor entre 0 e ∞ . A fórmula da conectividade [2] é dada por:

$$Conn(\Omega) = \sum_{i=1}^{N} \sum_{j=1}^{L} x_{i,t_{i(j)}}$$

Capítulo 3

Regressão *Clusterwise*

Este capítulo visa a oferecer uma ideia geral do que é a Regressão *Clusterwise*, explicando a definição do problema em que geralmente se usa, e também como funciona o modelo de regressão linear *clusterwise* assim como o modelo de regressão não-linear *clusterwise*. Será também mencionada a ideia de alocação de novas observações mostrando alguns métodos mais usados de alocação.

3.1 Definição do problema

Geralmente os dados que devem ser trabalhados na regressão linear são homogêneos, ou seja, a dispersão em relação a média é constante. A regressão *clusterwise* já vem sendo estudada há bastante tempo, com o intuito de segmentar o conjunto de dados em partições, ajustando o melhor modelo de regressão em cada partição, sendo utilizada quando há heterogeneidade nos dados. A regressão *clusterwise* pode ser vista como uma mistura particular ou latente de modelo de classe, sendo que numa perspectiva analítica de dados é vista como uma combinação de análise de agrupamento e regressão.

Existem muitos métodos de agrupamento que podem ser utilizados para a obtenção dos *clusters*, no entanto, este trabalho irá considerar apenas o *k-means*. Dentro do contexto de regressão, está sendo considerado que existe uma partição da variável resposta $Y = (Y_1, Y_2, ..., Y_c)$, com c sendo a quantidade de *clusters*/partições. Neste caso, tem-se para g $(1 \leq g \leq k)$ $Y_g = (y_{1(g)}, .., y_{n(g)})^T$ com n representado a quantidade de objetos/indivíduos, o mesmo ocorrendo em relação as variáveis independentes $X_1, X_2, ..., X_p$, sendo $X_g = (X_{1(g)}, ..., X_{p(g)})$ a matriz modelo associada a partição g, contendo as informações das respectivas p variáveis independentes, com $1 \leq g \leq c$. Assim, como no método K-*means*, o algoritmo de regressão *clusterwise* inicia-se de uma partição inicial dos dados, para um conhecido, realiza o ajuste do modelo de regressão em cada *cluster*, realiza a afetação das observações que são melhor ajustadas por um modelo de outro *cluster*, até convergência, ou seja, quando

nenhuma observação muda de *cluster*. A partição inicial influência bastante no resultado final do algoritmo, desta forma, ajuste-se o algoritmo considerando várias partições iniciais escolhendo aquela em que a função objetivo apresenta menor valor. A seguir apresenta-se, em maior detalhe, o método de regressão *clusterwise* considerando modelos de regressão linear e o caso não-linear.

3.2 Modelo de regressão linear *clusterwise*

No modelo de regressão *clusterwise* linear supõe-se que cada *cluster* contém, necessariamente, uma quantidade objetos que não pode ser menor que a quantidade de parâmetros do modelo. Sendo c o índice associado ao *cluster*, i o índice associado objeto, e j o índice associado ao atributo, o protótipo de cada *cluster* será definido por:

$$y_{i(k)} = \beta_{0(k)} + \sum_{j=1}^{p} \beta_{j(k)} x_{ij(k)} + \epsilon_{i(k)} \ (\forall i \in P_k, \text{ e } \#P_k \geq p).$$

O modelo de regressão *clusterwise* tem como objetivo minimizar uma função objetivo a cada iteração até convergência, sendo definida por:

$$J = \sum_{k=1}^{g} \sum_{i \in P_k} \epsilon_{i(k)}^2.$$

3.2.1 Estimação dos parâmetros

Para uma partição fixa composta de c *clusters*, é possível estimar os parâmetros dos c protótipos, aqui representados pelos respectivos modelos de regressão linear, utilizando o método dos mínimos quadrados. Seja o c-ésimo protótipo definido por:

$$\hat{y}_{i(c)} = \hat{\beta}_{0(c)} + \sum_{j=1}^{p} \hat{\beta}_{j(c)} x_{ij(c)} \ (\forall i \in P_c, \text{ e } \#P_c \geq p)$$

Seja P_c uma sub-amostra formada pelos elementos da partição c, as estimativas de mínimos quadrados $\hat{\beta}_{j(g)}$, $1, \leq g \leq c$, $0 \leq p \leq j$, o qual minimiza a função objetivo J, será obtido da seguinte operação matricial [4]:

$$\hat{\beta_{(c)}} = (\hat{\beta}_{0(c)}, \hat{\beta}_{1(c)}, ..., \hat{\beta}_{p(c)})^T = A_{(c)}^{-1} b_{(c)},$$

em que a matriz $A_{(c)}$ é quadrada e tem dimensão $(p+1) \times (p+1)$ e $b_{(c)}$ é um vetor com dimensão $(p+1) \times 1$, expressos por:

$$A_{(c)} = \begin{pmatrix} |P_c| & \sum_{i \in P_c} x_{i1} & \dots & \sum_{i \in P_c} x_{ip} \\ \sum_{i \in P_c} & \sum_{i \in P_c} x_{i1}^2 & \dots & \sum_{i \in P_c} x_{ip} x_{i1} \\ \vdots & \vdots & \vdots & \vdots \\ \sum_{i \in P_c} x_{ip} & \sum_{i \in P_c} x_{i1} x_{ip} & \dots & \sum_{i \in P_c} x_{ip} x_{ip}^2 \end{pmatrix}$$

$$b = (\sum_{i \in P_c} y_i, \sum_{i \in P_c} y_i x_{i1}, ..., \sum_{i \in P_c} y_i x_{ip})^T, \ \forall i \in P_c \text{ e } \#P_c \geq p.$$

3.2.2 Regra de Afetação

Uma vez que as estimativas dos parâmetros do modelo foram calculadas no passo da Estimação, o passo da Afetação consiste em re-definir os elementos que irão pertencer a partição c. Dessa forma, sabendo que E é uma lista de índices de todos os objetos, utilizou-se a seguinte regra para atualizar os elementos que irão pertencer a cada partição [4]:

$$P_c = \{i \in E : (\epsilon_{i(c)})^T \epsilon_{i(c)} \leq (\epsilon_{i(h)})^T \epsilon_{i(h)}, \forall h \neq c \ (h = 1, ..., g)\}$$

.

A estimativa para o erro $\epsilon_{i(c)}$ será dada pelo resíduo ordinário, expresso por:

$$r_{i(c)} = \hat{\epsilon}_{i(c)} = y_{i(c)} - (\hat{\beta}_{0(c)} + \sum_{j=1}^{p} \hat{\beta}_{j(c)} x_{ij(c)}) \ (\forall i \in P_c, \text{ e } \#P_c \geq p).$$

O símbolo # representa cardinalidade.

3.2.3 Algoritmo de regressão linear *clusterwise*

Dado um conjunto de dados com n elementos e número de *clusters* igual a c, uma condição necessária para o algoritmo de regressão linear *clusterwise* é que a quantidade de objetos em cada *cluster* seja pelo menos a quantidade de parâmetros do modelo, evitando a singularidade. A partir de uma partição inicial, será definido a amostra de modo a separar as observações por *clusters*. Na etapa seguinte, para cada *cluster* a estimação dos parâmetros do modelo, em seguida, é realizada a fase de afetação que servirá para mudar os objetos de um *cluster*/partição para o outro, de modo que se possa haver a redução da função objetivo J até a convergência. Para cada partição inicial haverá uma solução e a melhor das soluções será a que tiver o menor valor da função objetivo J. Um esquema para tal algoritmo pode ser visto na Figura 3.1 e no pseudocódigo apresentado no Algoritmo 1.

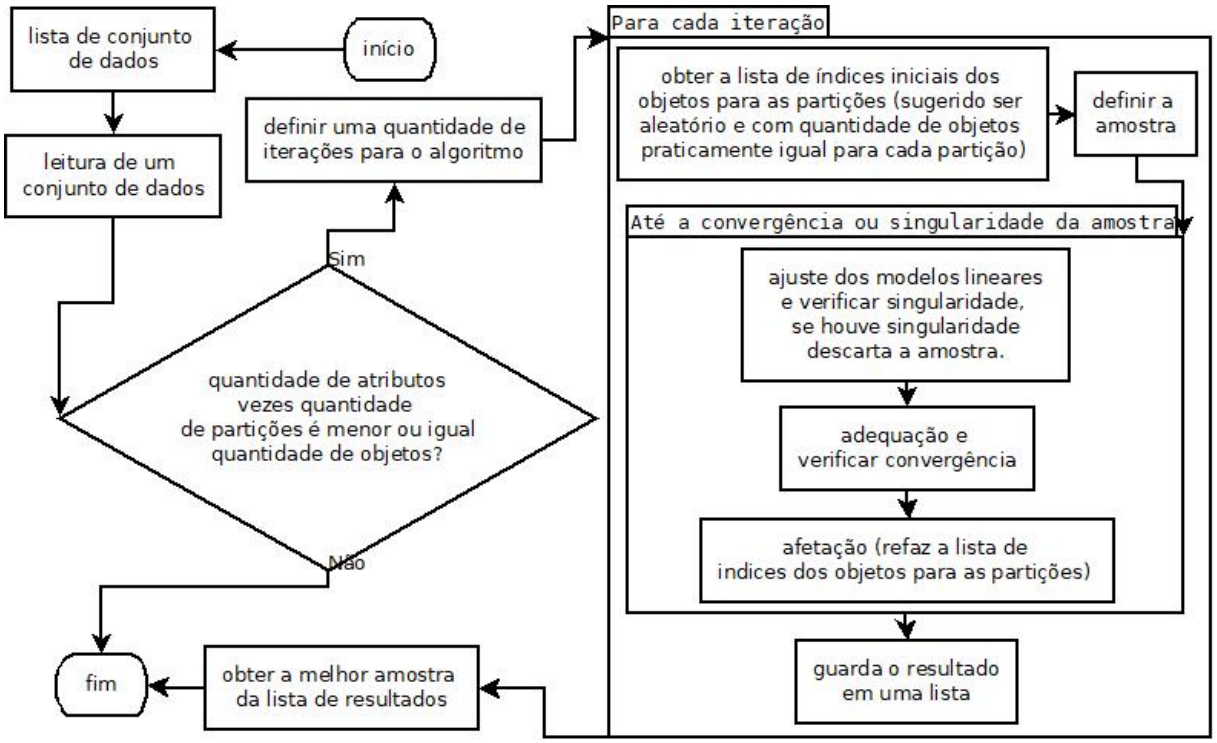

Figura 3.1: Esquema do algoritmo de regressão linear *clusterwise*

Autoria própria

Algoritmo 1 Regressão Linear *Clusterwise*

Entrada: $E = (e_1, \ldots, e_n)$, número de grupos c; representação dos objetos na forma $\mathcal{D} = \{(y_1, \mathbf{x}_1), \ldots, (y_1, \mathbf{x}_N)\}$

Saída: Uma lista dos modelos estimados (protótipos) e uma partição ótima $\mathcal{P} = (P_1, \ldots, P_c)$

Inicialização:
Defina $t \leftarrow 0$
Atribua aleatoriamente os objetos $\boldsymbol{e}_i$ para o *cluster* P_c para formar a partição inicial $\mathcal{P} = (P_1, \ldots, P_c)$;
Passo 1: Estimação
Defina $t \leftarrow t + 1$
Para $1 \leq c \leq C$ **Faça**
    Calcule estimativas dos parâmetros dos modelos lineares $\hat{\beta_{0(c)}}, \ldots, \hat{\beta_{p(c)}}$.
fim Para
Passo 2: Afetação
$\mathcal{P}^{(t)} = \mathcal{P}^{(t-1)}$;
$test \leftarrow 0$;
Para $1 \leq i \leq n$ **Faça**
    Seja $m : \boldsymbol{x}_i \in P_m^{(t)}$
    Encontre o *cluster* vencedor tal que

$$c = \arg\min_{1 \leq h \leq C} \left(y_{i(h)} - \hat{\beta}_{0(h)} - \sum_{j=1}^{p} x_{ij(h)} \hat{\beta}_{j(h)} \right)$$

    Se $c \neq m$ **Então**
        $test \leftarrow 1$
        $P_c = P_c \cup \{x_i\}$
        $P_m = P_m \setminus \{x_i\}$
    fim Se
fim Para
Compute o valor atual de J de acordo com a Equação $\sum_{c=1}^{C} \sum_{e_i \in P_c} \left(y_{i(c)} - \beta_{0(c)} - \sum_{j=1}^{p} x_{ij(c)} \beta_{j(c)} \right)^2$
Critério de parada:
Se $test == 0$ **Então**
    pare;
Senão
    $t \leftarrow t + 1$ e volte para o passo 1;
fim Se

Fonte: Adaptado da dissertação de CARVALHO *et al.* [5].

3.3 Modelo de regressão não-linear *clusterwise*

3.3.1 Regressão não-linear

Nem sempre um modelo de regressão linear traz uma boa aproximação em relação as variáveis preditoras e a variável resposta, por isso há a necessidade de se consi-

derar a possibilidade de uma relação não-linear entre as variáveis de interesse. O maior problema do modelo de regressão não-linear está no fato de que, nem sempre, as equações normais apresentam solução analítica, sendo necessário recorrer aos métodos iterativos de otimização.

A função que representa o modelo de regressão não-linear será definida por:

$$\Upsilon_u = f(\xi_u, \theta) + \epsilon_u,$$

onde u é a observação, $\xi_u = (\xi_{u1}, ..., \xi_{up})^t$ é o conjunto das variáveis preditoras, $\theta = \{\theta_1, ..., \theta_q\}$ é o vetor de parâmetros para do modelo e ϵ_u representa o erro aleatório. Assim como no caso linear, $E(\epsilon_u) = 0$ e, por consequência, $E(\Upsilon_u) = f(\xi_u, \theta)$, além do mais a $Var(\epsilon_u) = \sigma^2$[21].

A soma dos quadrados dos erros para o modelo de regressão não-linear é dada por:

$$S(\theta) = \sum_{u=1}^{n} \{\Upsilon_u - f(\xi_u, \theta)\}^2.$$

Para obter as estimativas do vetor de parâmetros $\hat{\theta}$ que minimizam $S(\theta)$ é necessário diferencia-lá em relação aos parâmetros e igualar a zero, obtendo o sistema de q equações normais:

$$\sum_{u=1}^{n} \{\Upsilon_u - f(\xi_u, \theta)\} \left[\frac{\partial f(\xi_u, \theta)}{\partial \theta_j}\right]_{\theta=\hat{\theta}} = 0, j = 1, \ldots, q.$$

Como o sistema de equações normais será não-linear, o que dificulta a obtenção da solução do mesmo, uma alternativa é linearizar $f(\xi_u, \theta)$ por Séries de Taylor e, em seguida, obter de forma iterativa a solução de um sistema linear através do método dos mínimos quadrados. Tal procedimento é conhecido como método de Newton-Raphson. Segundo [5], o primeiro passo é expandir $f(\xi_u; \theta)$ em séries de Taylor, até a primeira ordem, em torno de de um ponto $\theta_0 = \{\theta_{10}, ..., \theta_{p0}\}$, assim:

$$f(\xi_u, \theta) = f(\xi_u, \theta_0) + \sum_{i=1}^{p} \left[\frac{\partial f(\xi_u, \theta)}{\partial \theta_i}\right]_{\theta=\theta_0} (\theta_i - \theta_{i0}).$$

Define-se:

- $f_u^0 = f(\xi_u, \theta_0)$;
- $\beta_i^0 = \theta_i - \theta_{i0}$;
- $Z_{iu}^0 = \left[\frac{\partial f(\xi_u, \theta)}{\partial \theta_i}\right]$.

Reescrevendo a equação $\Upsilon_u = f(\xi_u, \theta) + \epsilon_u$, tem-se que:

$$\Upsilon_u - f_u^0 = \sum_{i=1}^{p} \beta_i^0 Z_{iu}^0 + \epsilon_u.$$

Colocando em notação matricial, tem-se:

$$y_0 = Z_0 \beta_0 + \epsilon,$$

onde:

$$Z_0 = \begin{bmatrix} Z_{11}^0 & Z_{21}^0 & \cdots & Z_{p1}^0 \\ Z_{12}^0 & Z_{22}^0 & \cdots & Z_{p2}^0 \\ \vdots & \vdots & & \vdots \\ Z_{1u}^0 & Z_{2n}^0 & \cdots & Z_{pu}^0 \end{bmatrix} \quad \hat{\beta}_0 = \begin{bmatrix} \hat{\beta}_1^0 \\ \hat{\beta}_2^0 \\ \vdots \\ \hat{\beta}_p^0 \end{bmatrix} \quad y_0 = \Upsilon - f^0 = \begin{bmatrix} \Upsilon_1 - f_1^0 \\ \Upsilon_2 - f_2^0 \\ \vdots \\ \Upsilon_u - f_u^0 \end{bmatrix}.$$

Uma vez linearizada, a estimativa para o vetor $\hat{\beta}_0$ é obtida como solução de um problema de mínimos quadrados, expressa por:

$$\hat{\beta}_0 = (Z_0^T Z_0)^{-1} Z_0^T (\Upsilon - f^0).$$

Sabendo que $\beta_0 = \theta - \theta_0$, define-se a estimativa revisada de θ como: $\hat{\theta}^v = \hat{\beta}^{v-1} + \theta^{v-1}$, com o indicador da iteração $v \geq 1$ e o momento de parada dado pelo erro relativo:

$$\left| \frac{\hat{\theta}_j^{v+1} - \hat{\theta}_j^v}{\hat{\theta}_j^v} \right| \leq \delta, \forall j = 1, \ldots q,$$

e δ um valor pequeno a ser escolhido.

Vários problemas podem advir ao método iterativo acima apresentado, dentre eles, dificuldade de convergência, sendo necessário um numero grande de iterações; pode haver flutuação até o momento da solução chegar à estabilidade; o método pode não convergir, fazendo com que a soma dos quadrados aumente em vez de diminuir [5].

3.3.2 Modelo de regressão não-linear *clusterwise*

Um modelo de regressão não-linear *clusterwise* foi proposto por CARVALHO *et al.* [5] para dados do tipo-intervalo, fazendo uso de um algoritmo de agrupamento dinâmico e dos modelos de regressão linear e não-linear. Será explanado tal algoritmo considerando que as amplitudes do intervalos são nulas, o que nos remete ao caso de dados usuais como caso particular.

A função objetivo J pode ser expressa por:

$$J = \sum_{c=1}^{g} \sum_{i \in P_c} \epsilon_{i(c)}^2 = \sum_{c=1}^{g} \sum_{i \in P_c} \left[y_{i(c)} - f_{(c)ho}(x_{i(c)}, \beta_{(c)}) \right]^2,$$

em que o índice h remete a uma família de funções não-lineares pré-definidas $(f_1, f_2, \ldots, f_h)$; o índice c remete ao *cluster* e o índice o remete ao método de otimização utilizado. Fixado o *cluster* h, a função h e a heurística de otimização o, obtém-se as estimativas dos parâmetros que minimiza a expressão abaixo:

$$\sum_{i=1}^{n} \left[y_{i(c)} - f_{(c)}(x_i, \beta_{(c)}) \right] \left[\frac{\partial f_{(c)}(x_i, \beta_{(c)})}{\partial \beta_{j(c)}} \right]_{\beta = \hat{\beta}} = 0.$$

O algoritmo proposto por CARVALHO *et al.* [5] considera as seguintes heurísticas de otimização: Simulating Annealing(SANN), Gradiente Conjugado (CG) e BFGS (Broyden–Fletcher–Goldfarb–Shanno).

A modificação do algoritmo de Regressão *Clusterwise* não-linear para dados tipo intervalo proposto por CARVALHO *et al.* [5], denotado por iCRCNLR (*Interval Clusterwise Non-Linear Regression*), é descrito no pseudocódigo do Algoritmo 2 e ilustrado na Figura 3.2, tem como objetivo apenas demonstrar como é um algoritmo de Regressão *Clusterwise* não-linear para dados usuais.

Algoritmo 2 Algoritmo iCRCNLR modificado

Entrada: $\mathcal{D} = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_n)$, número de *clusters* C, H modelos de funções candidatas e O heurísticas de otimização;

Saída: Uma lista contendo os melhores dentre os H modelos (protótipos) e uma partição ótima $\mathcal{P} = (P_1, \ldots, P_C)$.

Inicialização: Defina $t \leftarrow 0$ Atribua aleatoriamente os objetos $\boldsymbol{t}_i$, $i = 1, \ldots, n$, para o *cluster* P_c, $c = 1, \ldots, C$ para formar a partição inicial $\mathcal{P}^{(0)} = \left(P_1^{(0)}, \ldots, P_C^{(0)}\right)$;

Passo 1: Ajuste do Modelo

Defina $t \leftarrow t + 1$

Para $1 \leq c \leq C$ **Faça**

  Para $1 \leq h \leq H$ **Faça**

    Para $1 \leq o \leq O$ **Faça**

      Compute, utilizando o método de otimização iterativo o, $\hat{\boldsymbol{\beta}}_{(c)ho}$;

      Armazene as estimativas da primeira das O heurísticas que convergir e siga; Se nenhuma das O heurísticas convergir, armazene o último modelo e suas estimativas e siga;

    fim Para

  fim Para

  Selecione ho, tal que

$$f_{(c)ho} = \min_{1 \leq h \leq H} \sum_{e_i \in P_c} \left[y_i - f_{(c)h}(\mathbf{x}_i, \boldsymbol{\beta}_{(c)}) \right]^2$$

fim Para

Passo 2: Alocação

$\mathcal{P}^{(t)} = \mathcal{P}^{(t-1)}$;

$test \leftarrow 0$;

Para $1 \leq i \leq n$ **Faça**

  Seja $m : \boldsymbol{x}_i \in P_m^{(t)}$

  Encontre o *cluster* vencedor tal que

$$c = \arg\min_{1 \leq h \leq C} \left\{ \left[\left(\hat{\boldsymbol{\epsilon}}_{i(h)} \right)^{(t)} \right]^2 \right\}$$

  Se $c \neq m$ **Então**

    $test \leftarrow 1$, $P_c = P_c \cup \{\boldsymbol{x}_i\}$, $P_m = P_m \setminus \{\boldsymbol{x}_i\}$;

  fim Se

fim Para

Compute o valor atual de J de acordo com a equação $\sum_{c=1}^{C} \sum_{e_i \in P_c} \left(\boldsymbol{\epsilon}_{i(c)} \right)^{\top} \boldsymbol{\epsilon}_{i(c)}$

Critério de parada:

Se $test == 0$ **Então**

  pare;

Senão

  $t \leftarrow t + 1$ e volte para o passo 1;

fim Se

Fonte: Algoritmo de Regressão *Clusterwise* não-linear. Adaptado do algoritmo iCRCNLR proposto por CARVALHO *et al.* [5]

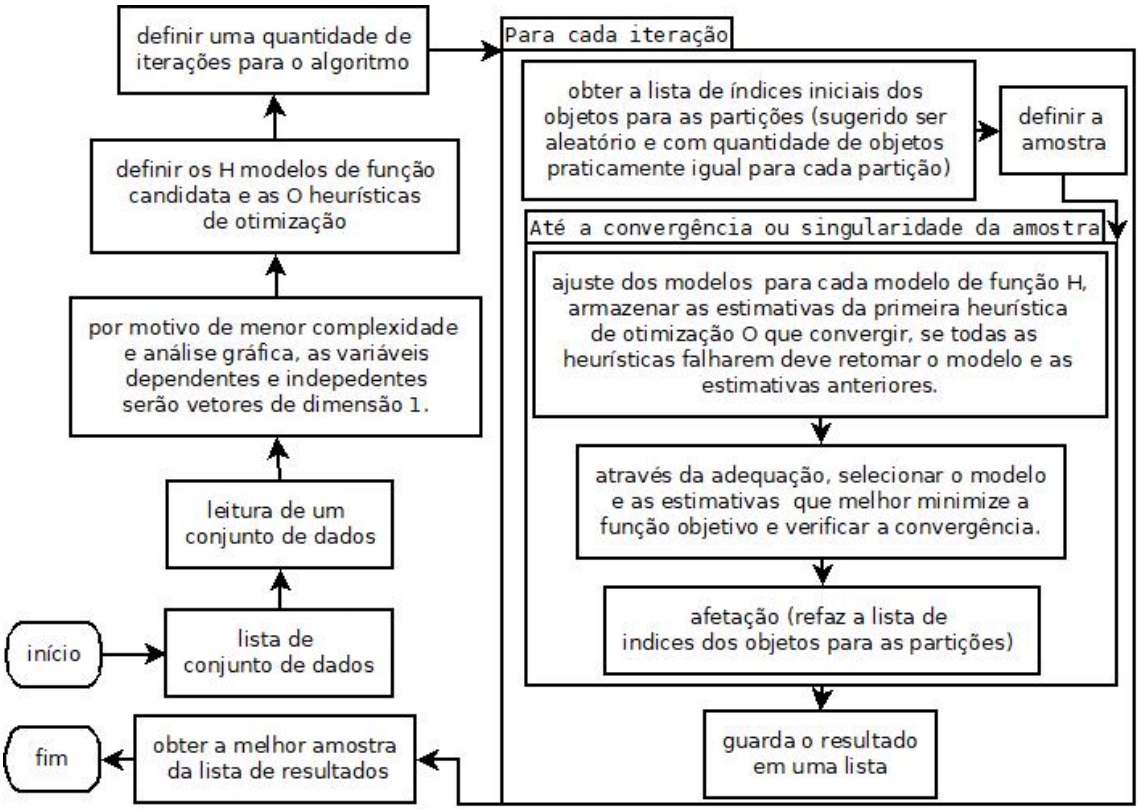

Figura 3.2: Modelo de Regressão *Clusterwise* não-linear

Autoria própria.

3.4 Alocação de Novas Observações

O ajuste de um modelo de regressão *clusterwise* e definição das observações que representam cada um dos *clusters*, que determina a convergência do algoritmo, representa uma parte do problema em questão. Em problemas práticos, a segmentação obtida servirá de referência para alocar novas observações, que não participaram da fase de estimação dos parâmetros dos modelos nem da fase de afetação. A seguir, discutirá algumas técnicas existentes na literatura que objetivam determinar qual o *cluster* mais adequado para alocar uma nova observação, com base nas informações fornecidas pelas variáveis independentes. Dentre as técnicas que serão apresentadas estão KNN, *Stacked Regression* e Alocação Aleatória.

3.4.1 Alocação aleatória

A maneira mais simples de se alocar uma nova observação seria escolher o *cluster* a qual a observação irá pertencer de forma aleatória. A probabilidade de uma nova observação pertencer a um *cluster* pode ser, por exemplo, de acordo com a quantidade de observações em cada *cluster*, ou ser igualmente provável para cada *cluster*.

3.4.2 Alocação com KNN

Outro método de alocação é conhecido como *K-Nearest Neighbors* (KNN) que nada mais é que um método de aprendizagem não-paramétrico baseado em instâncias que pode ser utilizado tanto para uma classificação quanto para uma regressão. Classificar determinado indivíduo pressupõe estimar a classe a qual pertence. No KNN, a classificação é feita por voto majoritário em relação as classes dos seus vizinhos próximos. A quantidade de vizinhos é vista como uma constante K e essa constante pode ser definida pelo usuário ou por validação cruzada. No entanto, o voto também pode ser ponderado com a distância do indivíduos a seus vizinhos. Para o cálculo da distância, utiliza-se uma determinada métrica, sendo a mais comum a distância euclidiana que é expressa por:

$$d(x, y) = d(y, x) = \sqrt{(x_1 - y_1)^2 + \ldots + (x_n - y_n)^2}.$$

No contexto da regressão *clusterwise* pode ser usado o KNN para a classificação de uma instância, alocando um determinado objeto a uma *cluster*/partição [5], assim:

1. A regressão *clusterwise* fornece as partições dos dados, com cada indivíduo pertencente a apenas um *cluster*.

2. Uma nova observação ou instância deverá ser alocada a classe com o KNN utilizando uma medida de distância em relação às variáveis independentes X, dado que a variável dependente Y não será observada no indivíduo a ser classificado;

3. Por fim, o *cluster* do novo indivíduo será estimado a partir dos votos majoritários dos k-vizinhos mais próximos.

3.4.3 Alocação com *Stacked Regression*

Em vez da escolha de apenas 1 *cluster* para a predição é possível fazer uma espécie de ponderação entre todos os *cluster* para assim fazer uma predição. Essa é a ideia básica do *Stacked Regression*, o qual tenta obter uma melhor acurácia das predições. Na regressão *clusterwise* a adaptação necessária é substituir os preditores pelos modelos obtidos em cada *cluster* pelo algoritmo, fazendo uma combinação linear das predições de cada *cluster* para assim obter uma predição. Será também útil a utilização da validação cruzada e o método de Mínimos Quadrados, tendo a restrição de não negatividade para assim estimar os coeficientes da combinação linear dos preditores. Disponha-se de C preditores dados por $v_1(x), \ldots, v_C(x)$, em relação a variável y, usando o vetor x [5]. Tendo em mente que o algoritmo quer

combinar os $v_1(\boldsymbol{x}), \ldots, v_C(\boldsymbol{x})$ para se obter uma maior acurácia, assim faz necessário reajustar os preditores utilizando o mesmo conjunto de dados de modo a excluir a n-ésima observação, para a isso tem-se a notação $v_C^{-n}(x)$, $z_n = (z_{n1}, z_{n2}, ..., z_{nk})$ é um vetor de C entradas com:

$$z_{cn} = v_c^{(-n)}(x_n)$$

Cria-se um novo conjunto de dados formado por $\{(y_n, z_n), n = 1, ..., N\}$. Pode-se fazer a escolha de apenas um preditor utilizando a minimização de $\sum_c y_n - z_{cn}$ para a escolha do c, no entanto há possibilidade de se fazer uma combinação entre preditores, para assim melhorar as acurácias da predição fazendo uso da seguinte equação:

$$v(x) = \sum_c \alpha_c v_c(x)$$

Os coeficientes α são obtidos ao minimizar a seguinte equação:

$$\sum_c (y_n - \sum_c \alpha_c v_c(x_n))^2$$

Para evitar o *overfit*, devido ao fato da aprendizagem ser no mesmo conjunto de dados, faz necessário o uso da validação cruzada, separando o que deve ser treinamento e predição do conjunto de dados. Modificando a equação anterior para:

$$\sum_c (y_n - \sum_c \alpha_c z_{cn})^2$$

Outro problema que deve ser mencionado é a alta correlação dos preditores devido ao fato de ter que predizer os mesmos valores. Sendo necessário o uso do método *ridge regression*, que por objetivo propõe a adição de uma restrição na minimização a qual é $\sum a_c^2 = \rho$, no qual s é o valor selecionado por validação cruzada.[5]

O passo a passo para o uso do *stacked regression* se dá da seguinte forma:

1. Particionar o conjunto de dados em L-*folds*;
2. Escolher um dos *folds* para ser utilizado para teste e os outros para treinamento;
3. Incluir uma outra validação cruzada somente para os dados de treino, ou seja, particionar o treino em L_{treino} *folds*;
4. Usar os $L_{treino} - 1$ *folds* para fazer o treinamento e prever os resultados do *fold* utilizado para a predição.
5. Para cada observação do *fold* para a predição, obter $\hat{y_{n1}}, \ldots, \hat{y_{nC}}, y_n$, $n = 1, \ldots, N_f$, em que N_f é o número de elementos dentro do *fold* para a predição;

6. Fazer o procedimento descrito para todos os $L_{treino}folds$ do conjunto de treinamento;

7. Com os dados obtidos em todos os *folds*, estimar $\boldsymbol{\alpha}$;

8. Fazer uso da equação $\sum_c (y_n - \sum_c \alpha_c z_{cn})^2$ para prever os valores do *fold* separado para teste no passo 2.

9. Fazer o procedimento para os $L - 1$ *folds* restantes do conjunto de dados.

Capítulo 4

Modelo de segmentação *clusterwise* com protótipos híbridos

Neste capítulo será apresentado o Modelo de Segmentação *Clusterwise* com Protótipos Híbridos (MoSCH), a prova de convergência, novas regras para alocação de novas observações, a escolha automática da quantidade de *cluster*, além de uma breve descrição dos modelos utilizados no algoritmo proposto.

4.1 Algoritmo de segmentação *clusterwise* com protótipos híbridos

Um fluxograma ilustrando o algoritmo de segmentação *clusterwise* com protótipos híbridos é apresentado na Figura 4.1. O algoritmo foi implementado em linguagem R, utilizando o RStudio. Dentre os métodos de aprendizagem e modelos estatísticos que foram considerados nesta proposta de dissertação como candidatos a protótipos MoSCH tem-se: SVR (*Support Vector Regression*), o qual ao contrário do SVM (Support Vector Machine) a predição é numérica, o GLM (*Generalized Linear Model*), KNN *regression*, Regressão Robusta, Árvores de Inferência Condicional e o Modelo Aditivo Generalizado. Portanto, o algoritmo engloba várias estruturas de regressão. Ademais, o algoritmo permite a inclusão de outras técnicas de predição numérica, tais como: Redes Neurais, Regressão *Kernel*, *Gradient Boosting*, entre outros.

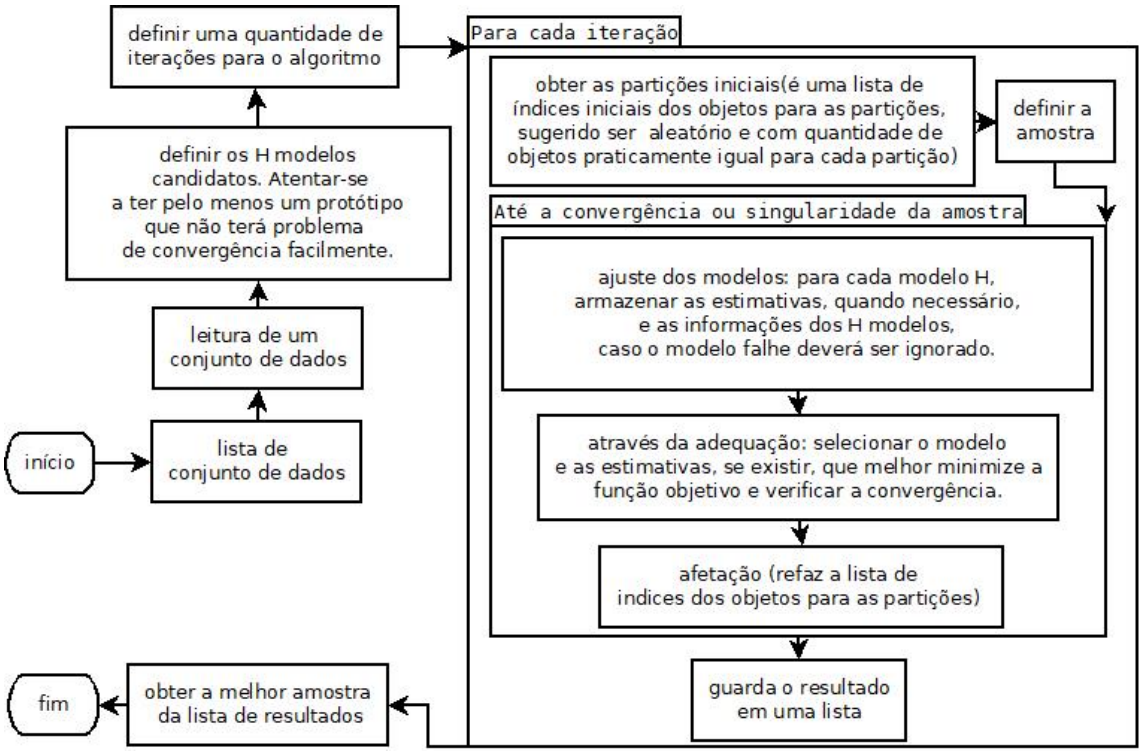

Figura 4.1: Fluxograma do algoritmo proposto

Autoria própria.

O Algoritmo 3 apresenta as etapas para implementação do algoritmo de segmentação *clusterwise* com protótipos híbridos. Inicialmente é necessário definir os objetos de entrada, o número de *clusters*, os possíveis modelos e o número máximo de iterações. O algoritmo inicia-se distribuindo os objetos em cada *cluster* de acordo com a partição inicial. Esta, por sua vez, pode seguir algum tipo de critério. O usado neste trabalho foi o aleatório, com uma quantidade de objetos aproximadamente igual em cada *cluster*. A cada passo, o algoritmo ajustará os H modelos em cada *cluster* e escolherá o modelo com menor soma dos quadrados dos resíduos. Caso um objeto, pertencente a um dado *cluster*, apresente uma melhor predição (menor resíduo) em função um modelo de um *cluster* diferente, este deverá ser afetado, o que implica em uma redução da função objetivo. Dessa forma, configura-se em uma nova partição dos dados e uma nova etapa de ajuste é realizada. Caso contrário, o algoritmo terminará a sua execução e retornará a lista dos melhores modelos e suas estimativas em cada *cluster*. Critérios inferências podem ser utilizados pelo especialista, quando aplicáveis, para validação das estimativas dos modelos.

Algoritmo 3 Algoritmo MoSCH

Entrada: $\mathcal{D} = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_n)$, número de *clusters* C, H modelos, P partição inicial, q numero máximo de iterações;

Saída: Uma lista contendo os melhores dentre os H modelos (protótipos) e suas estimativas, e uma partição ótima local $\mathcal{P} = (P_1, \ldots, P_C)$.

1: **Inicialização:** Defina $t \leftarrow 0$ Atribua os objetos $\boldsymbol{t}_i$, $i = 1, \ldots, n$, para o *cluster* P_C, $c = 1, \ldots, C$ de acordo com a partição inicial $\mathcal{P}^{(0)} = \left(P_1^{(0)}, \ldots, P_C^{(0)}\right)$;
2: **Passo 1:** Ajuste do Modelo
3: Defina $t \leftarrow t + 1$
4: **Para** $1 \leq c \leq C$ **Faça**
5: **Para** $1 \leq h \leq H$ **Faça**
6: Calcule e guarde para cada *cluster* as estimativas de todos os modelos em H ignorando as que falharem.
7: **fim Para**
8: Selecione h, tal que

$$f_{(c)h} = \min_{1 \leq h \leq H} \sum_{e_i \in P_c} \left[y_i - f_{(c)h}(\mathbf{x}_i, \boldsymbol{\beta}_{(c)}) \right]^2$$

9: **fim Para**
10: **Passo 2:** Alocação
11: $\mathcal{P}^{(t)} = \mathcal{P}^{(t-1)}$;
12: $test \leftarrow 0$;
13: **Para** $1 \leq i \leq n$ **Faça**
14: Seja $m : \boldsymbol{x}_i \in P_m^{(t)}$
15: Encontre o *cluster* vencedor tal que

$$c = \arg\min_{1 \leq h \leq C} \left\{ \left[\left(\hat{\boldsymbol{\epsilon}}_{i(h)} \right)^{(t)} \right]^2 \right\}$$

16: **Se** $c \neq m$ **Então**
17: $test \leftarrow 1$, $P_c = P_c \cup \{\boldsymbol{x}_i\}$, $P_m = P_m \setminus \{\boldsymbol{x}_i\}$;
18: **fim Se**
19: **fim Para**
20: Compute o valor atual de J de acordo com a equação $\sum_{c=1}^{C} \sum_{e_i \in P_c} \left(\boldsymbol{\epsilon}_{i(c)} \right)^\top \boldsymbol{\epsilon}_{i(c)}$
21: **Critério de parada:**
22: **Se** $test == 0$ ou $t == q$ **Então**
23: pare;
24: **Senão**
25: $t \leftarrow t + 1$ e volte para o passo 1;
26: **fim Se**

Fonte: Adaptado do algoritmo iCRCNLR proposto por CARVALHO *et al.* [5].

4.1.1 Prova de convergência do algoritmo proposto

O algoritmo proposto tem semelhanças com o proposto por CARVALHO *et al.* [5]. No entanto, o algoritmo MoSCH busca por uma partição $\mathcal{P} = (P_1, \ldots, P_C)$ de E em C *clusters* não vazios e um vetor de modelos C-dimensional $G = \left(m_{(1)}, \ldots, m_{(C)}\right)$,

em que $m_{(c)} = f_{(c)}(x)$ e $f_{(c)}$ é uma função que provem a estimativa do modelo $m_{(c)}$. Ainda o G e o $\mathcal{P}$ fornecem os valores que minimizam J:

$$J(\boldsymbol{G}, \mathcal{P}) = min\left\{J(\boldsymbol{G}, \mathcal{P}) : \boldsymbol{G} \in \mathbb{L}^C, \mathcal{U} \in \mathbb{P}_C\right\},$$

de tal forma que $\mathbb{P}_C$ é o conjunto de todas as partições de E em C *clusters* não vazio, tal que $P_c \in (p(E) - \emptyset)$, sendo que $p(E)$ é o conjunto de partes de E e $P_c \in \mathbb{P}$ e $\mathbb{L}$ representa o espaço de modelos contendo H modelos.

As propriedades de convergência deste tipo de algoritmo podem ser definidas a partir do estudo de duas séries, tomando algumas adaptações em relação ao apresentado por CARVALHO *et al.* [5], tem-se que: $v_t = \left(\boldsymbol{G}^t, \mathcal{P}_C^t\right) \in \mathbb{L}^C \times \mathbb{P}_C$ e $u_t = J(v_t) = J\left(\boldsymbol{G}^t, \mathcal{P}^t\right)$, $t = 0, 1, \ldots$. Partindo do termo inicial $v_0 = \left(\boldsymbol{G}^0, \mathcal{P}_C^0\right)$, o algoritmo proposto computa os termos da série até a convergência, em que o critério J assume um valor estacionário.

Assumi-se nas proposições a seguir que cada *cluster* ajustará um modelo como protótipo que convirja, mais informações podem ser vista na Subseção 4.5.

Proposition 1. *A série $u_t = J(v_t)$ diminui a cada iteração e converge.*

Demonstração. Primeiramente, demonstra-se que as desigualdades abaixo valem e decrescem a cada iteração:

$$J\left(\boldsymbol{G}^t, \mathbb{P}_C^t\right) \geq J\left(\boldsymbol{G}^{(t+1)}, \mathbb{P}_C^t\right) \geq J\left(\boldsymbol{G}^{(t+1)}, \mathbb{P}_C^{(t+1)}\right)$$

O lado esquerdo vale, uma vez que, para *f fixo*, a função selecionada $f_{(c)h}^{(t+1)}$ minimiza a seguinte soma de erros:

$$f_{(c)h}^{(t+1)} = \underset{1 \leq h \leq H}{\arg\min} \sum_{e_i \in P_c^t} \left[\boldsymbol{y}_i - f_{(c)h}(\boldsymbol{x}_i)\right]^2$$

Assim na iteração $t+1$, o valor da função objetivo reduz se comparado a iteração anterior.

$$\sum_{c=1}^{C} \sum_{e_i \in P_c} \left\{\left[\boldsymbol{y}_{i(c)}^t - f_{(c)h}^t\left(\boldsymbol{x}_{i(c)}^t\right)\right]\right\} \geq \sum_{c=1}^{C} \sum_{e_i \in P_c} \left\{\left[\boldsymbol{y}_{i(c)}^t - f_{(c)h}^{(t+1)}\left(\boldsymbol{x}_{i(c)}^t\right)\right]\right\}$$

Assim, uma vez que a função selecionada minimizam os erros,

$$J\left(\boldsymbol{G}^t, \mathcal{P}_C^{(t)}\right) \geq J\left(\boldsymbol{G}^{(t+1)}, \mathcal{P}_C^t\right)$$

Voltando a desigualdade

$$J\left(\boldsymbol{G}^t, \mathbb{P}_C^t\right) \geq J\left(\boldsymbol{G}^{(t+1)}, \mathbb{P}_C^t\right) \geq J\left(\boldsymbol{G}^{(t+1)}, \mathbb{P}_C^{(t+1)}\right)$$

O lado direito tem validade porque

$$\mathcal{P}_C^{(t+1)} = \underset{\mathcal{P}=(P_1,\ldots,P_C)\in\mathbb{P}_C}{\arg\min} \sum_{c=1}^{C} \sum_{e_i\in P_c} \left\{ \left[\boldsymbol{y}_{i(c)}^t - f_{(c)h}^t \left(\boldsymbol{x}_{i(c)}^t \right) \right] \right\}$$

Não há outra partição $\mathcal{P}$ que faça J decrescer mais do que $\mathbb{P}^{(t+1)}$, que é única. Com isso, tem-se que a segunda desigualdade do lado direito vale:

$$J\left(\boldsymbol{G}^{(t+1)}, \mathbb{P}_C^t\right) \geq J\left(\boldsymbol{G}^{(t+1)}, \mathbb{P}_C^{(t+1)}\right)$$

Assim, pode-se concluir que a série u_t decresce e é limitada ($J \geq 0$), e portanto, converge.

□

Proposition 2. *A série* $v_t = \left(\boldsymbol{G}^t, \mathcal{P}^t\right)$ *converge.*

Demonstração. Assumindo que $\mathcal{P}_t$ alcança estacionariedade na iteração T, $\mathcal{P}_T = \mathcal{P}_{(T+1)}$ e $J(v_T) = J(v_{(T+1)})$. Baseado na Proposição 1, tem-se que

$$J\left(\boldsymbol{G}^T, \mathcal{P}^T\right) = J\left(\boldsymbol{G}^{T+1}, \mathcal{P}^T\right) = J\left(\boldsymbol{G}^T, \mathcal{P}^{T+1}\right)$$

Para o lado esquerdo das igualdades, $\boldsymbol{G}^{(T+1)} = \boldsymbol{G}^{(T)}$. Para a partição fixada $\mathcal{P}^T$, as funções selecionadas são únicas e o critério de otimização J sustenta o mesmo valor. Para o lado direito da equação, $\mathcal{P}^{(T+1)} = \mathcal{P}^{(T)}$ ocorre porque $\mathcal{P}$ é única a cada iteração, ou seja, para $\boldsymbol{G}$ não há duas ou mais partições que minimizam J. □

4.2 Alocação de novos dados

A alocação de novas observações representa um aspecto importante em problemas de *clusterwise regression*. É um procedimento que visa estimar em qual *cluster* um novo objeto deve ser associado. Alguns métodos de alocação utilizados em problemas de regressão clusterwise são: KNN (Seção 3.4.2), *Stacked Regression* (Seção 3.4.3) e a Alocação aleatória (Seção 3.4.1). Dessa forma, o algoritmo MoSCH também será avaliado em relação a performance em problemas de alocação de novas observações utilizando os métodos de alocação citados.

Em relação ao *Stacked regression* em vez da utilização do *Leave-One-Out* foi utilizado o Q-*Fold*, e as estimativas dos seus coeficientes podem ser obtidas de duas formas: (i) considerando que os coeficientes devem ser valores não negativos (será chamado de StkRegSCINN) e (ii) considerando que os coeficientes devem ser não negativos e a soma de todos os coeficientes deve ser 1 (será chamado de StkRegSCI1). Este último (StkRegSCI1) configura-se como uma contribuição deste trabalho.

Uma variação para alocação através da técnica de KNN também foi proposta, o qual será chamada de KNN dos *clusters* combinados (será chamada de AKNC). Nela, faz-se o uso de uma média ponderada inversamente proporcional a distância dos K vizinhos próximos, em que:

1. k será a quantidade de vizinhos próximos.
2. X será uma matriz bidimensional das variáveis independentes, em que cada linha X_i representa uma observação. A quantidade de observações deve ser maior que k.
3. D será um vetor das distâncias em que cada linha corresponde a distância da nova observação a todas as observações já existentes $D_i = \sqrt{\sum_j^p (X_{ij} - r_j)^2}$
4. I são os k primeiros índices da ordenação crescente das distâncias em D;
5. PP é um vetor de índices de todos os *clusters*;
6. G_i é o *cluster* de X_i, sendo G_i um índice diferente de vazio;
7. $M_{G_i \in PP}(r)$ é a predição do modelo do *cluster* de X_i para uma nova observação r;
8. $S = \sum_{i \in I} D_i$

Portanto, sabendo que $S = 0$ é pouco provável, a fórmula da predição do KNN dos *clusters* combinados para uma nova observação r será a média ponderada das predições dos *clusters*, associados aqueles vizinhos próximos, sendo o peso dado pela diferença entre a soma das distâncias de todos os vizinhos e a distância daquele vizinho próximo:

$$AKNC = \begin{cases} \frac{\sum_{i \in I}(S - D_i) M_{G_i \in PP}(r)}{\sum_{i \in I}(S - D_i)} & \text{se } S \neq 0 \\ \sum_{i \in I} M_{G_i \in PP}(r)/k & \text{se } S = 0 \end{cases}$$

4.3 Seleção automática da quantidade de *clusters*

Os algoritmos de *clusterwise* não fazem uso de uma escolha automática da quantidade de *clusters*. No entanto, esse tipo de procedimento pode ser bastante útil quando não se pretende executar o método várias vezes atê se descobrir a quantidade ideal de *clusters*.

O Algoritmo 4 visa descobrir a melhor quantidade de *clusters* para o método K-*Means*, apesar de poder ser utilizado para qualquer outro método. Inicialmente, é necessário ter como entrada a matriz de dados com X e Y, e também um valor para

a tolerância que será usada como um suporte para se descobrir a melhor quantidade de *clusters*. Considerou-se 0,1 para a tolerância. Depois, é necessário definir o número máximos de *clusters* que podem ser usados. Esse valor pode depender da quantidade de observações. Em seguida, definimos como métrica comparativa o REQM (Raiz do Erro Quadrático Médio) para cada valor da quantidade de *clusters* e normaliza-se em relação ao maior valor do REQM encontrado. Depois será feito o módulo da diferença entre o valor do REQM normalizado (referente a quantidade de *cluster*) atual e o anterior. Caso essa diferença seja menor que a tolerância, armazena-se o valor da quantidade de *clusters* atual como a melhor quantidade e para-se o algoritmo.

Muitos algoritmos de *clusterwise* não fazem uso de uma escolha automática da quantidade de *clusters*, assim o Algoritmo 4 foi implementado de modo a sugerir um valor de K a partir do método K-*Means*, sendo esse K ótimo local para o K-*Means* e o K sugerido é usado no algoritmo proposto. Pode-se afirmar que utilizar o Algoritmo 4 pode não trazer a melhor escolha da quantidade de *clusters* para o algoritmo proposto, mas também não será a pior escolha. Na comparação entre os algoritmos baseados no K-*Means* pode ser uma boa abordagem pois evita algum tipo de tendência ou preferência humana, e também o K-*Means* geralmente é um método menos custoso.

Algoritmo 4 Estimando o número de *cluster* C para algoritmo K-*Means*

Entrada: $XY = (x_1, \ldots, x_n, y)$; valor entre 0 e 1 para a *tolerancia*;
Saída: Valor estimado da melhor quantidade de grupos, $melhorC$

Inicialização:
Defina m como o número de linhas da matriz XY
Defina $ceiling(valor)$ como uma função que arredonda um valor para cima
Defina $cs \leftarrow 1 : ceiling(\sqrt{(m)})$
Defina $tamCs$ como o tamanho do vetor cs
Defina $melhorC \leftarrow cs[1]$
Defina $REQMs$ como vetor decimal de tamanho m
Passo 1: Obter o REQM de cada quantidade de grupo
Para $1 \leq c \leq tamCs$ **Faça**
    Calcule o REQM a partir do K-*Means* para cada quantidade de grupo em cs.
    $Erros \leftarrow Kmeans(dados = XY, centro = c)\$erros$
    $REQMs[c] \leftarrow \sqrt{(\sum(Erros^2))}$
fim Para
Passo 2: Normalização do REQM pelo seu maior valor
$normREQMs \leftarrow \frac{REQMs}{max(REQMs)}$;
Passo 3: Obter o melhor c com base na tolerância
Para $2 \leq c \leq tamCs$ **Faça**
    Critério de parada:
    Se $|normREQMs[c] - normREQMs[c-1]| < tolerancia$ **Então**
        $melhorC \leftarrow c$
        pare;
    fim Se
fim Para

Fonte: Autoria própria

4.4 Modelos utilizados no algoritmo proposto

Nesta seção tem-se uma breve apresentação dos modelos utilizados nessa dissertação para o algoritmo MoSCH. Conforme comentado anteriormente, o algoritmo proposto não se limita aos métodos aqui apresentados, podendo ser modificado e ampliado a critério do especialista.

4.4.1 Modelo linear generalizado (MLG)

O *generalized linear model* (GLM) é uma flexível generalização da regressão de mínimos quadrados ordinária. Foram propostos por John Nelder e Robert Wedderburn em 1972 no intuito de unificar vários modelos estatísticos. Alguns casos especias desta classe de modelos são: binomial com ligação logit, gaussiana com ligação identidade, Gamma com ligação inversa, gaussiana inversa com ligação $1/mu^2$, poisson com ligação log, quasi com ligação identidade e variancia constante, quasibinomial com ligação logit, quasipoisson com ligação log.

4.4.2 Regressão por vetores de suporte (RVS)

O *support vector regression* (SVR) é um conceito que é usado em análise de regressão, possuindo um conjunto de métodos de aprendizado supervisionado que tem como objetivo analisar os dados e reconhecer padrões. O que o SVR faz é obter uma separação, adotada por uma função *Kernel*, sendo mais comumente usado para a predição numérica. Algumas funções *Kernel* desse modelo são: linear, polinomial, radial e sigmoid.

4.4.3 Modelo aditivo generalizado (MAG)

O *Generalized additive model* (GAM) é uma extensão do modelo linear generalizado (MLG) incorporando estruturas suavizadoras entre as variáveis independentes. Assim, sua diferença está em denotar uma função não paramétrica a qual é estimada por meio de curvas de alisamento, não sendo necessário assumir uma relação linear entre $g(\mu_i)$ e as variáveis explicativas.

4.4.4 KNN de regressão

O KNN *regression* é um método que usa a média dos vizinhos mais próximos para estimar o valor da variável dependente a partir das variáveis independentes. Algo que pode ser feito em relação aos dados é um pré-processamento.

4.4.5 Árvores de inferência condicional

O *Conditional Inference Trees* (CTree) é um método que usa divisões univariadas da variável dependente, a partir dos valores das covariáveis. Apesar de terem um propósito parecido, não confundir com arvore de decisão, pois o Ctree utiliza um procedimento que faz uso de teste de significância (testes de permutação) para escolher variáveis, em vez de obter a variável que maximiza uma determinada medida de informação.

4.4.6 Regressão Robusta

A principal característica da *Robust Regression* é não utilizar os mínimos quadrados ordinários para as previsões, pois este método apresenta uma sensibilidade a valores atípicos. O uso da Regressão Robusta ocorre quando os dados contêm valores discrepantes, ou seja, a presença de *outliers*. Existem vários estimadores para a regressão robusta como por exemplo: Mínimos quadrados ponderados, M-Estimador e o MM-estimador.

4.5 Limitações

A seguir, descreve-se algumas limitações que o algoritmo MoSCH apresentam:

- A depender da escolha das técnicas utilizadas no algoritmo MoSCH, existe a possibilidade de alguns dos modelos utilizados como protótipos não convergir. Um problema maior ocorrerá se todos os modelos não convergirem. Entretanto, técnicas como o KNNReg tendem a não ter problemas de convergência. Ter ao menos um modelo com tal característica é recomendável pois o *cluster* poderia ajustar pelo menos um protótipo.

- Para alguns protótipos, principalmente os de regressão, a quantidade mínima de observações em cada *cluster* deve ser maior ou igual a quantidade de variáveis, de modo a evitar problemas na estimação dos modelos.

- Deve-se considerar uma quantidade razoável de partições iniciais para atingir-se uma boa convergência, avaliasse a razoabilidade de acordo com a circunstância. As partições inciais podem ser definidas de forma aleatória ou utilizando algum algoritmo. A performance do algoritmo MoSCH depende da partição inicial, não havendo garantias de se obter um mínimo global ao minimizar-se a função objetivo.

- Por ser robusto precisando de um considerável custo computacional e por conter inúmeros modelos os quais também podem ou não ser onerosos, o método MoSCH ele geralmente necessita de muito tempo para entregar um resultado, para este trabalho foi na casa de dias.

Em relação a alocação de novas observações, vale a pena destacar:

- Os problemas podem ocorrer no método de alocação *Stacked Regression*. Ao reestimar um modelo de determinado *cluster* no conjunto de treinamento, pode-se ter problemas de convergência uma vez que o modelo não pode ser alterado. Para o calculo dos coeficientes no StkRegSCI1, realiza-se uma decomposição de Cholesky para depois resolver o problema de *quadratic programming*. Assim , faz-se necessária uma matriz definida positiva, caso não a tenha, calculada-se uma matriz definida próxima. No cálculo destes coeficientes pode ocorrer outros problemas como se ter uma matriz degenerada ou uma matriz singular.

Capítulo 5

Análise Experimental

Neste capítulo é investigado o comportamento do algoritmo para o Modelo de Segmentação *Clusterwise* com Protótipos Híbridos (MoSCH), em comparação com o algoritmo de regressão *clusterwise* linear (RCL). Será utilizada como métrica para comparação o *Root Mean Square Error* (REQM). Foram realizadas simulações considerando 6 diferentes cenários, cada um representando uma estrutura de *clusterwise* diferente. Estes cenários foram construídos de acordo com as seguintes características: com ou sem sobreposição/interseção entre os clusters; tipo de estrutura de regressão no *cluster* (linear ou não-linear) e o número de *clusters* $g = \{2, 3\}$.

5.1 Recursos Computacionais

5.1.1 Especificações técnicas dos computadores

Para os dados simulados foram utilizadas máquinas fornecidas pelo Departamento de Estatística, tendo cada uma a configuração Processador Intel Xeon Silver 4114, 2.2GHz 20 núcleos/40 threads, 64 GB RAM, 64bits, Manjaro Linux.

5.1.2 Linguagem `R`

`R` é um ambiente de *software* livre para computação e gráficos estatísticos distribuído sob um *copyleft* no estilo GNU e uma parte oficial do projeto GNU ("GNU S"). Foi inicialmente escrita por Ross Ihaka e Robert Gentleman, do Departamento de Estatística da Universidade de Auckland, em Auckland, Nova Zelândia. Atualmente, há uma vasta contribuição de pacotes em linguagem `R`, possuindo uma comunidade significativa. O nome `R` vem da letra inicial dos criadores. Possui suporte a várias plataformas como Unix, MacOs e Windows. O *design* do `R` teve influência em duas linguagens: S e Scheme. O núcleo `R` é uma linguagem para computador interpretada que permite ramificações e *loop*, assim como programação modular usando

funções. As funções, na maioria das vezes, são escritas pela própria linguagem, mas é possível haver funções escritas nas linguagens C, C++ ou Fortran, por meio de uma interface para obtenção de maior eficiência. A linguagem R possui uma vasta possibilidade de uso de procedimentos estatísticos, dentre estes: modelos lineares e lineares generalizados, modelos de regressão não-linear, análise de séries temporais, testes paramétricos e não-paramétricos, análise de agrupamento, entre outras. Além disso, tem um ambiente gráfico versátil para uma boa apresentação dos dados. Módulos adicionais também estão disponível para muitas finalidades específicas.

O RStudio é uma IDE (*Integrated Development Environment*) feita na linguagem C++ e utiliza o *framework* Qt, tendo a missão de fornecer um *software* profissional de código aberto, sendo amplamente usado para o ambiente de computação estatística `R`. O RStudio e a linguagem `R` foram utilizados neste trabalho para elaboração e implementação dos algoritmos de regressão *clusterwise* existentes, bem como do método de segmentação *clusterwise* com protótipos híbridos. Além disso, a fase experimental bem como a aplicação a dados reais também serão desenvolvidos na linguagem `R`.

Mais informações sobre a linguagem `R` e o RStudio podem ser encontradas em R-PROJECT [28], CRAN [7] e RSTUDIO [31].

5.2 Configurações utilizadas

Foram utilizados 20 diferentes modelos no algoritmo MoSCH para ajuste em cada *cluster*, conforme descritos no Capítulo 4. A seguir será apresentado as principais configurações dos modelos considerados no algoritmo de segmentação *clusterwise* com protótipos híbridos (MoSCH) e utilizados nesta seção:

GLM

Foi considerada a função `glm` do pacote `stats` do software `R v3.6.3` para ajuste dos GLMs de interesse. No total, foram considerados os seguintes modelos com suas respectivas ligações:

1. Gaussiano com ligação identidade;
2. Gaussiano com ligação log;
3. Gaussiano com ligação inversa;
4. Gamma com ligação identidade
5. Gamma com ligação log;

6. Gamma com ligação inversa;

7. Gaussiana inversa com ligação identidade;

8. Gaussiana inversa com ligação log;

9. Gaussiana inversa com ligação inversa.

SVR

Foi considerada a função `svm` do pacote `e1071 v1.7-4` do software `R v3.6.3` a qual possui suporte também para regressão. Para o ajuste dos SVRs de interesse foram considerados os seguintes modelos:

1. *kernel* polinomial de grau 3 com regressão nu;

2. *kernel* radial de grau 3 com regressão nu;

3. *kernel* sigmoid de grau 3 com regressão nu;

4. *kernel* polinomial de grau 3 com regressão eps;

5. *kernel* radial de grau 3 com regressão eps;

6. *kernel* sigmoid de grau 3 com regressão eps;

Em todas as configurações, considerou-se uma tolerância de 0,001, custo = 1, cache de tamanho = 40 e a padronização em relação a média e variância.

GAM

Foi considerada a função `gam` do pacote `mgcv v1.8-33` do software `R v3.6.3` para ajuste dos GAMs de interesse. Nosso objetivo aqui foi ajustar modelos de regressão não-paramétrica aos dados. Dessa forma, nenhuma variável independente foi considerada na estrutura paramétrica do modelo.

As configurações em relação aos modelos utilizados com GAM:

1. utilizar o método REML para o GAM e na formula cada atributo das variáveis independentes tem um *smooth* (função `s` do pacote `mgcv`) com parâmetros k=2, bs='cr', fx = TRUE;

2. utilizar o método REML para o GAM e na formula cada atributo das variáveis independentes tem um *smooth* (função `s` do pacote `mgcv`) com parâmetros k=2, bs='gp', fx = TRUE;

KNN *regression*

Foi considerada a função `train` do pacote `caret v6.0-86` do software `R v3.6.3` para ajuste dos modelos com KNN. As configurações utilizadas no KNN *regression* foram:

1. controle de treino (gerado pela função `trainControl`) utilizando o método repeatedcv com 3 repetições;
2. pré-processamento utilizando centro e escala;
3. utiliza a seguinte fórmula para determinar uma lista de valores de k para serem testado no KNN:
   (a) $kmax$ é a raiz quadrada da quantidade de observações;
   (b) $stepk$ é 2 vezes o arredondamento para cima de $kmax/20$, tem como objetivo gerar por volta de 10 valores de k;
   (c) $listaks$ é uma lista de valores de k gerada por uma sequencia iniciada com valor 3 com passo dado por $stepk$ e com valor máximo $kmax$.

Ctree

Foi considerada a função `ctree` do pacote `party v1.3-6` do software `R v3.6.3` para ajuste dos modelos com Ctree. Não houve necessidade de nenhum tipo de configuração além da padrão para o modelo Ctree.

Regressão Robusta

Foi considerada a função `lmrob` do pacote `robustbase v0.93-7` do software `R v3.6.3` para ajuste dos modelos com Regressão Robusta. Para a Regressão Robusta foi definido o estimador MM.

5.3 Experimentos com dados sintéticos

Na Tabela 5.1 apresenta as configurações dos dados sintéticos. A função $\mathfrak{U}(a, b, c)$ é uma função a qual gera uma distribuição uniforme e a função $\mathfrak{N}(a, b, c)$ é uma função que gera uma distribuição normal, onde o termo a remete a quantidade de dados que serão gerados, já o termo b é a média da distribuição e o termo c é o desvio-padrão.

Tabela 5.1: Configurações dos dados sintéticos para cada cenário, com tamanho de amostra em cada *cluster* de $n = 100$

Cenário	*Cluster*	Variável Explicativa	Erro	Variável Resposta
1	1	$X_{c1} = \mathfrak{U}(n, 0, 10)$	$e_1 = \mathfrak{N}(n, 0, 8)$	$Y_{c1} = 10 * X_{c1} + e_1$
	2	$X_{c2} = \mathfrak{U}(n, 7, 17)$	$e_2 = \mathfrak{N}(n, 0, 8)$	$Y_{c2} = 230 - 10 * X_{c2} + e_2$
2	1	$X_{c1} = \mathfrak{U}(n, 0, 10)$	$e_1 = \mathfrak{N}(n, 0, 6)$	$Y_{c1} = 100/(1 + exp(-1.5 * (X_{c1} - 5))) + e_1$
	2	$X_{c2} = \mathfrak{U}(n, 8, 18)$	$e_2 = \mathfrak{N}(n, 0, 6)$	$Y_{c2} = 220 - 10 * X_{c2} + e_2$
3	1	$X_{c1} = \mathfrak{U}(n, 0, 10)$	$e_1 = \mathfrak{N}(n, 0, 16)$	$Y_{c1} = 300/(1 + exp(-1.5 * (X_{c1} - 5))) + e_1$
	2	$X_{c2} = \mathfrak{U}(n, 8, 18)$	$e_2 = \mathfrak{N}(n, 0, 16)$	$Y_{c2} = 300 + 300/(1 + exp(-0.95 * (X_{c2} - 9))) + e_2$
4	1	$X_{c1} = \mathfrak{U}(n, 0, 4)$	$e_1 = \mathfrak{N}(n, 0.0, 1.0)$	$Y_{c1} = 4 + 1 * X_{c1} + e_1$
	2	$X_{c2} = \mathfrak{U}(n, 3, 6)$	$e_2 = \mathfrak{N}(n, 0.0, 1.0)$	$Y_{c2} = -8 + 3 * X_{c2} + e_2$
	3	$X_{c3} = \mathfrak{U}(n, 5, 8)$	$e_3 = \mathfrak{N}(n, 0.0, 1.0)$	$Y_{c3} = 10 - 3 * X_{c3} + e_3$
5	1	$X_{c1} = \mathfrak{U}(n, 0, 4)$	$e_1 = \mathfrak{N}(n, 0.0, 0.3)$	$Y_{c1} = 2 + 1 * X_{c1} + e_1$
	2	$X_{c2} = \mathfrak{U}(n, 2, 9)$	$e_2 = \mathfrak{U}(n, 0.01, 0.7)$	$Y_{c2} = X_{c2}^{0.75-1} * exp(-X_{c2}/(-4)) + e_2$
	3	$X_{c3} = \mathfrak{U}(n, 1, 10)$	$e_3 = \mathfrak{U}(n, 0.01, 0.7)$	$Y_{c3} = X_{c3}^{0.75-1} * exp(-X_{c3}/(-3.5)) + e_3$
6	1	$X_{c1} = \mathfrak{U}(n, 2, 7)$	$e_1 = \mathfrak{U}(n, 0.01, 0.15)$	$Y_{c1} = X_{c1}^{4-1} * exp(-X_{c1}/(1)) + e_1$
	2	$X_{c2} = \mathfrak{U}(n, 0, 7)$	$e_2 = \mathfrak{U}(n, 0.01, 0.15)$	$Y_{c2} = X_{c2}^{2-1} * exp(-X_{c2}/(2)) + e_2$
	3	$X_{c3} = \mathfrak{U}(n, 0, 7)$	$e_3 = \mathfrak{U}(n, 0.01, 0.15)$	$Y_{c3} = X_{c3}^{1-1} * exp(-X_{c3}/(3)) + e_3$

Fonte: Autoria própria.

Na Figura 5.1 ilustra-se os cenários que podem ser gerados. É possível perceber que nos cenários 1, 2 e 3 há baixa sobreposição. Já nos cenários 4, 5 e 6 há uma considerável sobreposição, sendo que o cenário 5 e 6 também tem uma intersecção entre os *clusters*. Todos os cenários foram gerados com pequena dispersão, de modo os modelos não sejam afetados por uma variabilidade elevada dos dados. Note também que o cenário 1 é formado por 2 *clusters* lineares, o cenário 2 é formado por 1 *cluster* linear e 1 não-linear, o cenário 3 é formado por 2 *clusters* não-lineares, o cenário 4 é formado por 3 *clusters* lineares, o cenário 5 é formado por 1 *cluster* linear e 2 não-lineares. Por fim, o cenário 6 é formado por 3 *clusters* não-lineares.

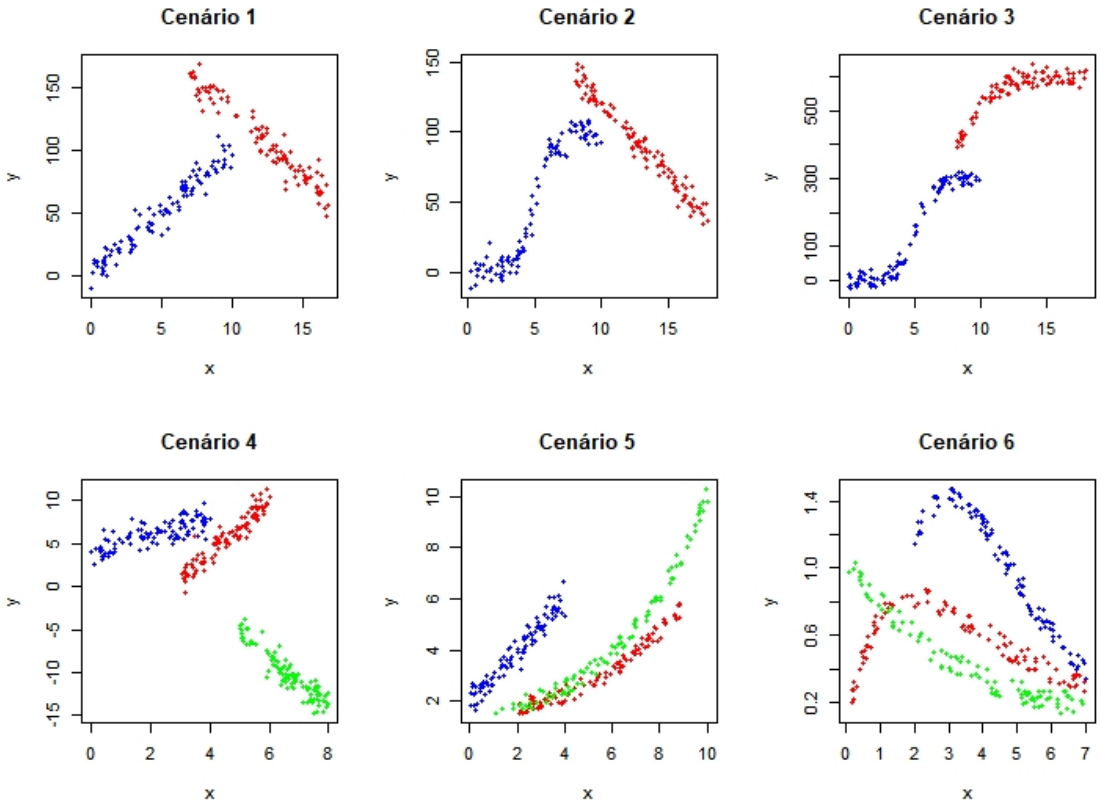

Figura 5.1: Cenários para os dados sintéticos

Autoria própria.

5.4 Resultados para os experimentos com dados sintéticos

A Figura 5.2 apresenta o procedimento para obter os resultados encontrados na Tabela 5.2. Foram utilizados dois algoritmos para comparação, o RCL e o MoSCH. Em cada cenário os algoritmos tiveram as mesmas entradas, diferindo apenas em relação aos modelos que são usados e nos valores das partições iniciais geradas. O MoSCH utiliza todos os modelos apresentados na Seção 5.2, enquanto que o RCL utiliza apenas o GLM gaussiano com ligação identidade. As entradas semelhantes em cada cenário são: a variável dependente, a variável independente, a quantidade de *clusters*, a quantidade de iterações máxima e a quantidade de partições iniciais.

Pela visão geral fornecida pela Figura 5.2 é necessário instanciar as matrizes para armazenar os valores REQM, tanto para o algoritmo RCL como para o MoSCH. Em seguida, fará uso do Método de Monte Carlo o qual foi definido com 500 iterações. Em cada iteração de Monte Carlo serão executados 6 cenários. Em cada cenário são gerados os dados sintéticos para cada *cluster* (C={2,3}), com tamanho de amostra em cada *cluster* igual a 100. Os algoritmos farão uso da variável dependente e da

variável independente, considerando as configurações apresentadas na Tabela 5.1. Utilizou-se 30 partições iniciais e a quantidade máxima de iterações igual a 100. Os algoritmos fornecerão o valor SQR para cada *clusters*, que por sua vez serão usados para o cálculo do REQM que deverá ser armazenado na Matriz REQM do respectivo algoritmo em uma dada iteração de Monte Carlo e cenário. Ao terminar todas as iterações de Monte Carlo, as matrizes com os valores de REQM de determinado algoritmo servirão para o cálculo do REQM médio e do teste de hipóteses em de cada cenário.

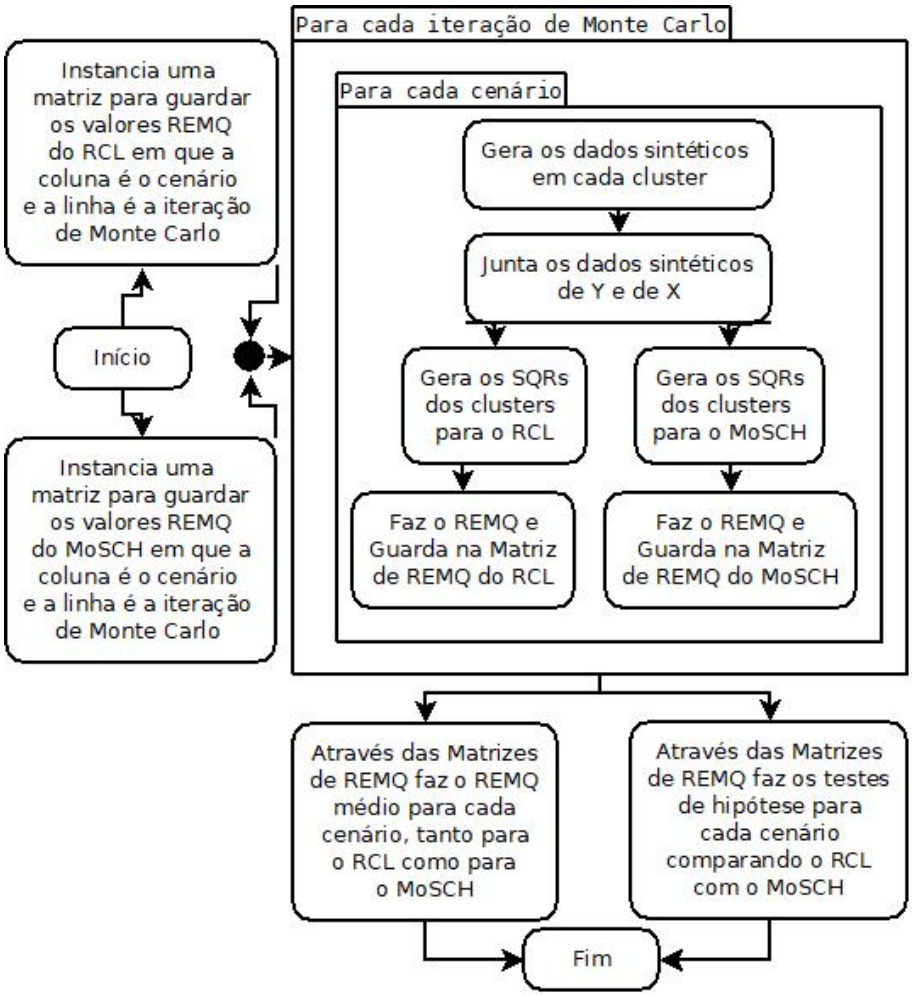

Figura 5.2: Algoritmo para gerar os resultados dos dados sintéticos
Autoria própria.

A Tabela 5.2 apresenta os resultados dos algoritmos mostrando o REQM médio dos ajustes, em cada cenário, incluindo também o teste de Wilcoxon. A partir do p-valor encontrado é perceptível verificar que o MoSCH apresentou um menor REQM médio, em todos cenários, quando comparado com o RCL. Dessa forma, os resultados corroboram que o algoritmo MoSCH apresentou uma excelente performance, quando comparado com o RCL.

Tabela 5.2: Comparação entre os algoritmos MoSCH e RCL, por cenário. REQM e p-valor do teste de Wilcoxon para comparação de medianas.

Cenário						
Método	1	2	3	4	5	6
MoSCH	5,0242	3,5390	10,0416	0,5605	0,1762	0,0480
	(0,4584)	(0,3021)	(1,0839)	(0,0968)	(0,0411)	(0,0060)
RCL	7,7166	10,8390	36,2151	0,9350	0,3176	0,0949
	(0,3880)	(0,5257)	(1,3307)	(0,0455)	(0,0383)	(0,0055)
p-valor	$6,34e^{-84}$	$6,34e^{-84}$	$6,34e^{-84}$	$1,12e^{-83}$	$6,34e^{-84}$	$6,38e^{-84}$
H_0:MoSCH = RCL						
H_1:MoSCH < RCL						

Fonte: Autoria própria.

Capítulo 6

Aplicações a Dados Reais

Neste capítulo serão apresentadas aplicações do algoritmo proposto a bases de dados reais. Além disso, o mesmo será comparado com algumas abordagens presentes na literatura. Para tanto, as principais configurações adotadas nos algoritmos foram as mesmas da Seção 5.2, com exceção que foi utilizado um notebook para executar as simulações com dados reais e para outros trabalhos convencionais inerentes, com configuração Core i7-7500, 2.7GHz, 16Gb RAM, 64bits, Windows 10, Nvídea Geforce 940MX 4GB. Serão descritos os modelos, sua performance em termos de ajuste, bem como o poder preditivo na alocação de novas observações. Também serão avaliados os diferentes métodos de alocação em conjuntos de teste, por meio do esquema de validação cruzada. A comparação entre os diferentes algoritmos e métodos de alocação dar-se-a por meio de testes de hipóteses não-paramétricos (Wilcoxon).

Destaca-se que o algoritmo proposto (MoSCH) será comparado com o algoritmo de regressão *clusterwise* linear (RCL), o método K-*Means* seguido do ajuste de regressões lineares em cada cluster e o método K-*Means* seguido do ajuste do melhor protótipo híbrido para cada *cluster*. Estas duas últimas abordagem servirão de *baseline* para evidenciar o ganho devido ao uso de uma estrutura de regressão *clusterwise* ao problema.

6.1 Considerações metodológicas

Inicialmente, cada base de dados considerada nessa seção passou por um processo de pré-tratamento, onde foram removidas linhas com valores inválidos, organizada as posições das colunas e realizada as tabulações necessárias nas variáveis independentes categóricas, que precisavam ser definidas como fatores. Em seguida, tem-se a etapa de definição da quantidade de *clusters* utilizando o algoritmo apresentado na Seção 4.2.

Por último, são executados os algoritmos RCL, MoSCH, K-*Means* linear e K-*Means* híbrido. Cada algoritmo requer como entrada as variáveis dependente e

independentes, a lista de modelos, a quantidade de *clusters*, o número máximo de iterações. A partir de uma mesma partição inicial, os algoritmos iniciam o processo de ajuste de modo a otimizar sua função objetivo. Como todos esses métodos são influenciados pela partição inicial, esse processo foi repetido considerando 30 partições iniciais. O resultado apresentado para cada algoritmo retrata o menor valor da função objetivo obtida considerando todas as partições iniciais.

Uma vez ajustado os algoritmos, uma segunda parte do estudo consiste em avaliar o poder preditivo dos métodos para novas observações, bem como avaliar qual o melhor método de alocação de novas observações. Para tanto, utiliza-se uma validação cruzada 10-*fold* em que 9 *folds* são usados como conjunto de treinamento e o *fold* restante como conjunto de teste. Os métodos de alocação são então aplicados e sua performance avaliada em termos da raiz do erro quadrático médio (REQM).

Configurações dos métodos de alocação

A seguir, serão apresentadas algumas informações importantes que foram consideradas nos 5 métodos de alocação utilizados nesta dissertação:

1. Alocação aleatória: selecionará um *cluster* aleatoriamente entre os retornados por algum algoritmo e fará a predição com base no modelo e estimativas associados a ele.

2. KNN de classificação: selecionará um *cluster* entre os retornados por algum algoritmo pelo voto majoritário dos vizinhos mais próximos e fará a predição com base no modelo e estimativas associados ao *cluster*. O método utiliza um *grid* de valores para k (número de vizinhos), gerado da seguinte forma:

   (a) $kmax$ é a raiz quadrada da quantidade de observações no *fold*;

   (b) $stepk$ é 2 vezes o arredondamento para cima de $kmax/20$ e tem por objetivo gerar cerca de 10 valores para k;

   (c) $listaks$ é uma lista de valores de k gerada por uma sequencia iniciada com valor 3 e passo dado por $stepk$, com valor máximo $kmax$.

3. A alocação com KNN dos *clusters* combinados: fará a predição usando a predição de todos os *clusters* retornados por algum algoritmo, através de uma média ponderada pelas distâncias entre os vizinhos mais próximos. O método utiliza uma lista de valores para k, gerada da seguinte forma:

   (a) $kmax$ é a raiz quadrada da quantidade de observações no *fold*;

   (b) $stepk$ é 2 vezes o arredondamento para cima de $kmax/20$ e tem como objetivo gerar por volta de 10 valores de k;

(c) *listaks* é uma lista de valores de k gerada por uma sequencia iniciada com valor 3, com passo dado por *stepk*, e com valor máximo *kmax*.

4. *Stacked regression*: Fará uso do *Stacked regression* mencionado na Seção 3.4.3.

5. *Stacked regression* com a soma dos coeficiente igual a 1: Fará uso do *Stacked regression* mencionado na Subseção 3.4.3, porém considerando que a soma dos coeficiente seja igual a 1.

6.2 Bases de Dados Reais

Este trabalho foi subsidiado pelo programa FAPESQ - Fundação de Apoio à Pesquisa do Estado da Paraíba, em um projeto de pesquisa com aplicações associadas ao tema de Internet das Coisas (IoT). Dessa forma, algumas bases de dados utilizadas estão relacionadas ao tema. Por outro lado, o algoritmo proposto permite ser aplicado qualquer problema que envolva clusterização e regressão.

Todas as bases de dados utilizadas neste estudo foram obtidas do seguinte endereço eletrônico `https://blog.datasciencedojo.com/30-datasets-to-uplift-your-skills-in-data-science/` ou podem ser encontradas no seguinte repositório: `https://code.datasciencedojo.com/datasciencedojo/datasets`.

Por fim, considerar-se-á as seguintes nomenclaturas para os métodos de alocação: Alocação aleatória (AAle), Alocação com KNN de classificação (AKNN), Alocação com KNN dos *clusters* combinados (AKNC), Alocação com StkRegSCI1 (Stk1), Alocação com StkRegSCINN (StkN).

6.2.1 Conjunto de dados de eficiência energética

Esse conjunto de dados possui, originalmente, 768 observações e 10 variáveis. A variável *Load Ratio* foi criada e representa a razão entre *Heating Load* e *Cooling Load*, sendo tais variáveis não consideradas neste estudo.

Neste problema, o objetivo é avaliar a carga de aquecimento e requisitos de carga de resfriamento de edifícios (ou seja, eficiência energética) em função dos parâmetros do edifício. A base contém 12 formas diferentes de edifícios simuladas no Ecotect, a qual é uma ferramenta da Autodesk que oferece uma variedade de simulação e funcionalidade de análise da energia de edifício tendo o intuito de melhorar o funcionamento de edifícios já existentes e novos. Os edifícios diferem em relação à área envidraçada, à distribuição da área envidraçada e à orientação, entre outros parâmetros. A seguir, na Tabela 6.1, a descrição das variáveis para este conjunto de dados:

Tabela 6.1: Dados de eficiência energética: descrição das variáveis

Variável	Nome	Descrição	Tipo
Y	Load Ratio	Razão entre cargas	quantitativo
X1	Relative Compactness	Compacidade Relativa	quantitativo
X2	Surface Areas	Área da superfície	quantitativo
X3	Wall Area	Área da parede	quantitativo
X4	Roof Area	Área do telhado	quantitativo
X5	Height	Altura	quantitativo
X6	Orientation	Orientação	quantitativo
X7	Glazing Area	Área envidraçada	quantitativo
X8	Glazing Area Distribution	Distribuição da área envidraçada	quantitativo

Fonte: Adaptado de *Energy Efficiency Data Set*. Disponível em: <`https://code.datasciencedojo.com/datasciencedojo/datasets/tree/master/Energy%20Efficiency`>.

Ajuste dos algoritmos

A seguir, serão apresentados os resultados dos algoritmos a partir da função objetivo J mencionada na Subseção 3.2, que faz uso do cálculo da soma dos SQE em cada *cluster*.

Na Tabela 6.2 é possível perceber que o algoritmo RCL apresentou o melhor ajuste, seguido pelo algoritmo MoSCH. Mesmo partindo de partições iniciais iguais, o MoSCH possibilita o ajuste de outros modelos de regressão podendo, em algumas situações, apresentar um ajuste inferior ao RCL. Já o K-*Means* Híbrido superou o K-*Means* Linear. No K-*Means* híbrido apresentou, em 4 *clusters*, o modelo KNN e em 1 *cluster* o modelo SVM. O algoritmo MoSCH apresentou, em 2 *clusters*, o modelo KNN e em 3 *clusters* o modelo SVM. Nota-se que tanto o K-*Means* como o MoSCH escolheram os mesmos tipos de modelos.

Tabela 6.2: Comparação entre os algoritmos. Base de dados de eficiência energética utilizando 100 partições iniciais. Número de *cluster* = 5.

Algoritmo	Função Objetivo	Modelos
K-*Means*-Linear	3,1255	5 GLM (gaussiana: identidade);
RCL	0,1005	5 GLM (gaussiana: identidade);
K-*Means*-Híbrido	1,8089	3 KNN (k=3), 1 KNN (k=5), 1 SVM (*kernel* radial e regressão nu);
MoSCH	0,1160	2 KNN (k=3), 3 SVM (*kernel* radial e regressão nu);

Fonte: Autoria própria

Avaliação preditiva por validação cruzada

A seguir, serão apresentados os resultados da avaliação do melhor método de alocação no problema em questão. Detalhes de como foi realizada a validação cruzada podem ser encontrados na Subseção 6.1.

A Tabela 6.3 traz o REQM médio e o desvio padrão em relação aos 10 *folds* considerados na validação cruzada. O melhor método de alocação foi o método da alocação com KNN dos *clusters* combinados, pois apresentou as menores médias e desvios padrão. O pior foi o método da alocação aleatória, tendo as maiores médias e desvios-padrão. Vale destacar que a alocação com KNN dos *clusters* combinados foi uma contribuição deste trabalho. O melhor algoritmo foi o K-*Means* linear, que apresentou a menor média e o menor desvio-padrão e o pior algoritmo foi o RCL que apresentou maiores médias e desvios-padrão entre os algoritmos.

Um fato interessante é que, ao considerar o método AKNC, a diferença não foi tão evidente, pois, visualmente, as médias e os desvios padrão se apresentaram em valores bem próximos, podendo até mesmo ser visto dentro de uma margem de tolerância um empate na maioria dos casos.

Adotando um ranqueamento com base na menor média do REQM, com base nas informações apresentadas na Tabela 6.3, tem-se: o K-*Means* Híbrido (1º), MoSCH (2º), K-*Means* Linear (3º) e RCL (4º). O resultado final do ranqueamento está apresentado na Seção 6.3.

Tabela 6.3: Conjunto de dados de eficiência energética. Comparação entre os algoritmos, por método de alocação: Média e Desvio-Padrão do REQM no conjunto Teste. Validação Cruzada 10-*fold*, utilizando 30 partições iniciais.

Algoritmo	AAle	AKNN	AKNC	Stk1	StkN
K-*Means* Linear	0,0673 (0,0056)	0,0669 (0,0043)	**0,0655** (0,0047)	0,0659 (0,0049)	0,0678 (0,0060)
RCL	0,1050 (0,0228)	0,0910 (0,0079)	0,0792 (0,0084)	**0,0740** (0,0059)	0,1009 (0,0223)
K-*Means* Híbrido	0,0685 (0,0079)	0,0699 (0,0077)	**0,0586** (0,0066)	0,1076 (0,1376)	0,0619 (0,0069)
MoSCH	0,0816 (0,0134)	0,0898 (0,0177)	**0,0644** (0,0083)	0,0815 (0,0226)	0,0645 (0,0112)

Fonte: Autoria própria

Foram realizados testes de hipóteses relacionando os métodos de alocação para cada algoritmo, os quais podem ser encontrados no Apêndice A.2. Um resumo dos resultados dos testes de hipóteses apresentados no Apêndice A pode ser encontrada na Seção 6.4. De acordo com os testes de hipótese do Apêndice A.2, os melhores

métodos de alocação nesta base de dados foram: o AKNC para o K-*Means* Linear; o Stk1 para o RCL; o AKNC para o K-*Means* Híbrido; e o AKNC juntamente com o StkN para o MoSCH.

Também foram realizados testes de hipóteses relacionando os algoritmos, os quais podem ser encontrados no Apêndice B.2. As amostras utilizadas nos testes de hipóteses que comparam os algoritmos correspondem ao método de alocação que apresentou o menor REQM médio. De acordo com os testes de hipóteses do apêndice B.2 o melhor algoritmo nesta base de dados foi o K-*Means* Linear.

6.2.2 Conjunto de dados Auto MPG

Esse conjunto de dados possui originalmente 398 linhas (observações) e 9 colunas, no entanto a coluna *car name* e a coluna *origin* não foram consideradas, pois não possuem informações relevantes para a análise. A fonte original dessa base de dados é o *StatLib*, um arquivo de dados da Carnegie Mellon University, disponível em `http://lib.stat.cmu.edu/datasets/`.

O objetivo do uso dessa base de dados é prever a eficiência energética de um determinado carro, ou seja, o quanto esse carro gasta de combustível por meio da quilometragem, potência, ano do modelo e outras especificações técnicas. A seguir, na Tabela 6.4 tem-se o dicionário de dados do conjunto de dados.

Tabela 6.4: Dados de auto MPG: descrição das variáveis

Variável	Nome	Descrição	Tipo
Y	mpg	Eficiência de combustível medida em milhas por galão (mpg)	quantitativo
X1	cylinders	Número de cilindros do motor	quantitativo
X2	displacement	Deslocamento do motor (em polegadas cúbicas)	quantitativo
X3	horsepower	Cavalos-força domotor	quantitativo
X4	weight	Peso do veículo(em libras)	quantitativo
X5	acceleration	Tempo para acelerar de 0 a 60mph (em segundos)	quantitativo
X6	model year	Ano modelo	qualitativo

Fonte: Adaptado de *Auto MPG Data Set*. Disponível em: <`https://code.datasciencedojo.com/datasciencedojo/datasets/tree/master/Auto%20MPG`>.

Ajuste dos algoritmos

A seguir, serão apresentados os resultados dos algoritmos a partir da função objetivo J mencionada na Subseção 3.2, que faz uso do cálculo da soma dos SQE em cada

cluster.

Na Tabela 6.5 é possível perceber que o algoritmo MoSCH apresentou o melhor ajuste, seguido pelo algoritmo RCL. Já o K-*Means* Híbrido foi superior ao K-*Means* Linear. Ambos os algoritmos, K-*Means* híbrido e MoSCH, apresentaram o modelo do GAM em 1 *cluster*, o modelo SVM em 2 *clusters* e o modelo KNN em um 1 *cluster*.

Tabela 6.5: Comparação entre os algoritmos. Base de dados de auto MPG utilizando 100 partições iniciais. Número de *cluster* = 4.

Algoritmo	Função Objetivo	Modelos
K-*Means*-Linear	390.017,8	4 GLM (gaussiana: identidade);
RCL	43.770,3	4 GLM (gaussiana: identidade);
K-*Means*-Híbrido	222.039,8	1 GAM (gaussiana: identidade e bs = gp), 1 SVM (*kernel* radial e regressão nu), 2 KNN (k=3);
MoSCH	29.361,28	1 GAM (gaussiana: identidade e bs = gp), 2 SVM (*kernel* radial e regressão nu), 1 KNN (k=3);

Avaliação preditiva por validação cruzada

A seguir, serão apresentados os resultados da avaliação do melhor método de alocação para o conjunto de dados em análise. Detalhes de como foi realizada a validação pode ser encontrado na Subseção 6.1.

A Tabela 6.6 apresenta os valores da média e do desvio padrão REQM em relação aos 10 *folds* considerados na validação cruzada. Uma inspeção nessa tabela evidencia que o melhor método de alocação foi a alocação com o KNN dos *clusters* combinados pois apresentou as menores médias e os menores desvios padrão, enquanto que o método que apresentou o pior desempenho foi o da alocação com KNN de classificação, apresentando as maiores médias e os maiores desvios padrão. Novamente, é importante destacar que o método da alocação com KNN dos *clusters* combinados uma contribuição deste trabalho. O melhor algoritmo foi o MoSCH, que apresentou a menor média e o menor desvio padrão do REQM entre todos os algoritmos, enquanto que o pior algoritmo foi o K-*Means* linear, que apresentou a maior média e o maior desvio padrão do REQM entre os algoritmos.

Adotando um ranqueamento pela menor média do REQM, com base nas informações apresentadas na Tabela 6.6, tem-se: o MoSCH (1º), RCL (2º), K-*Means* Híbrido (3º) e K-*Means* Linear (4º). O resultado final do ranqueamento está apresentado na Seção 6.3.

Tabela 6.6: Conjunto de dados auto MPG, comparação entre os algoritmos, por método de alocação: Média e Desvio-Padrão do REQM no conjunto Teste. Validação Cruzada 10-*fold*, utilizando 30 partições iniciais.

Algoritmo	AAle	AKNN	AKNC	Stk1	StkN
K-*Means* Linear	40,1112 (13,2660)	41,2204 (12,5564)	**36,2956** (8,6031)	40,3266 (12,4543)	40.8047 (12.6915)
RCL	55.2907 (12.7276)	43.1838 (10.9751)	**34.2685** (4.8505)	47.6494 (9.3045)	53.4842 (14.0821)
K-*Means* Híbrido	40.8202 (18.4633)	38.5509 (18.9925)	**35.5918** (15.7291)	67.8580 (39.0313)	37.1840 (15.9449)
MoSCH	37.2888 (6.7244)	36.5953 (7.6382)	34.3257 (13.1084)	58.0764 (34.5385)	**31.5241** (6.2763)

Fonte: Autoria própria

Foram realizados testes de hipóteses relacionando os métodos de alocação para cada algoritmo, os quais podem ser encontrados no Apêndice A.3. Um resumo dos resultados dos testes de hipóteses apresentados no Apêndice A pode ser encontrada na Seção 6.4. De acordo com os testes de hipótese do Apêndice A.3 os melhores métodos de alocação foi o AKNC para o K-*Means* Linear e RCL, e o StkN para o K-*Means* Híbrido e MoSCH.

Também foram realizados testes de hipóteses relacionando os algoritmos, os quais podem ser encontrados no Apêndice B.3. As amostras utilizadas nos testes de hipóteses que comparam os algoritmos correspondem ao método de alocação que apresentou o menor REQM médio. De acordo com os testes de hipóteses do apêndice B.3 o melhor algoritmo foi o RCL.

6.2.3 Conjunto de dados de consumo de álcool

Esse conjunto de dados possui originalmente 345 linhas (observações) e 7 colunas, no entanto foi removida a coluna *Selector* que representava apenas uma partição das observações em um conjunto de treinamento e outro de teste. Os dados foram coletados pela *BUPA Medical Research Ltd* e estão disponíveis em `https://archive.ics.uci.edu/ml/datasets/Liver+Disorders`.

As informações nessa base de dados são provenientes de exames de sangue e o objetivo neste trabalho está em estimar a quantidade de bebida alcoólica ingerida diariamente quantizado a meio litro através de 5 outras variáveis. A seguir na Tabela 6.7 tem-se uma descrição das variáveis contidas nesse conjunto de dados.

Tabela 6.7: Dados de consumo de álcool: descrição das variáveis

Variável	Nome	Descrição	Tipo
Y	Drinks	Número de equivalentes de meio litro de bebidas alcoólicas consumidas por dia	quantitativo
X1	Mcv	Volume Corpuscular Médio (teste de sangue)	quantitativo
X2	Alkphos	Fosfotase alcalina (teste de sangue)	quantitativo
X3	Sgpt	Alamina aminotransferase (teste de sangue)	quantitativo
X4	Sgot	Aspartato Aminotransferase (exame de sangue)	quantitativo
X5	Gammagt	Gama-glutamiltranspeptidase (teste de sangue)	quantitativo

Fonte: Adaptado de *Liver Disorders Data Set*. Disponível em: <`https://code.datasciencedojo.com/datasciencedojo/datasets/tree/master/Liver%20Disorders`>.

Ajuste dos algoritmos

A seguir, serão apresentados os resultados dos algoritmos a partir da função objetivo J mencionada na Subseção 3.2, que faz uso do cálculo da soma dos SQE em cada *cluster*.

A Tabela 6.8 apresenta os valores da média e do desvio padrão do REQM em relação aos 10 *folds* considerados na validação cruzada. É possível perceber que o algoritmo MoSCH apresentou o melhor ajuste entre todos os algoritmos, seguido pelo algoritmo RCL, enquanto que o pior ajuste foi apresentado pelo algoritmo K-*Means* Linear. O algoritmo K-*Means* Híbrido apresentou um melhor ajuste apenas em comparação com o K-*Means* Linear. O K-*Means* híbrido apresentou o modelo SVM em 2 *clusters*, o modelo GAM em 1 *cluster* e o modelo KNN em 1 textitcluster, enquanto que o algoritmo MoSCH apresentou o modelo SVM em 3 *clusters* e o modelo GAM em 1 *cluster*.

Tabela 6.8: Comparação entre os algoritmos. Base de dados de consumo de álcool utilizando 100 partições iniciais. Número de *cluster* = 4.

Algoritmo	Função Objetivo	Modelos
K-*Means*-Linear	2.707,733	4 GLM (gaussiana: identidade);
RCL	204,1434	4 GLM (gaussiana: identidade);
K-*Means*-Híbrido	1.782,42	1 GAM (gaussiana: identidade e bs=gp), 1 SVM (*kernel* radial e regressão nu), 1 SVM (*kernel* radial e regressão eps), 1 KNN(k=3);
MoSCH	167,9702	1 GAM (gaussiana: identidade e bs=gp), 1 SVM (*kernel* radial e regressão eps), 2 SVM (*kernel* polinomial de grau 3 e regressão eps);

Fonte: Autoria própria

Avaliação preditiva por validação cruzada

A seguir, serão apresentados os resultados da avaliação do melhor método de alocação para o conjunto de dados em análise. Detalhes de como foi realizada a validação cruzada podem ser encontrados na Subseção 6.1.

A Tabela 6.9 apresenta os valores da média e do desvio padrão do REQM em relação aos 10 *folds* considerados na validação cruzada. Pode-se observar que o melhor método de alocação foi a alocação com o KNN de classificação pois apresentou as menores médias e os menores desvios padrão, ao passo que o pior foi o método da alocação StkRegSCI1, que apresentou as maiores médias e os maiores desvios padrão. O melhor algoritmo foi o K-*Means* Linear, que apresentou a menor média e menor desvio padrão entre todos os algoritmos, e o algoritmo que apresentou o pior desempenho foi o MoSCH, com a maior média e o maior desvio padrão do REQM.

Adotando um ranqueamento pela menor média do REQM, com base nas informações apresentadas na Tabela 6.9, tem-se: o K-*Means* Linear (1º), MoSCH (2º), K-*Means* Híbrido (3º) e RCL (4º). O resultado final do ranqueamento estará na Seção 6.3.

Tabela 6.9: Conjunto de dados de consumo de álcool, comparação entre os algoritmos, por método de alocação: Média e Desvio-Padrão do REQM no conjunto Teste. Validação Cruzada 10-*fold*, utilizando 30 partições iniciais.

Algoritmo	AAle	AKNN	AKNC	Stk1	StkN
K-*Means* Linear	3,8226 (1,2182)	3,2666 (0,6650)	**3,1130** (0,6370)	3,9002 (1,2112)	3,6921 (1,1928)
RCL	5,0731 (2,1806)	4,0895 (0,4927)	**3,9044** (0,5725)	4,8322 (1,4328)	4,6151 (1,5836)
K-*Means* Híbrido	4,5660 (3,6397)	**3,6140** (0,4685)	5,9265 (7,4406)	26,1046 (40,8063)	9,5274 (12,2655)
MoSCH	8,9855 (10,7710)	9,1672 (10,6285)	10,9519 (11,0038)	6,3178 (5,9381)	**3,5058** (0,7046)

Fonte: Autoria própria

Foram realizados testes de hipóteses relacionando os métodos de alocação para cada algoritmo, os quais podem ser encontrados no Apêndice A.4. Um resumo dos resultados dos testes de hipóteses apresentados no Apêndice A pode ser encontrada na Seção 6.4. De acordo com os testes de hipótese do Apêndice A.4 os melhores métodos de alocação foi o AKNC para os algoritmos K-*Means* Linear, RCL e K-*Means* híbrido, e StkN para o algoritmo MoSCH.

Também foram realizados testes de hipóteses relacionando os algoritmos, os quais podem ser encontrados no Apêndice B.4. As amostras utilizadas nos testes de hipó-

teses que comparam os algoritmos correspondem ao método de alocação que apresentou o menor REQM médio. De acordo com os testes de hipóteses do apêndice B.4 o melhor algoritmo foi o MoSCH.

6.2.4 Conjunto de dados de preços de casas

Esse conjunto de dados possui originalmente 414 linhas (observações) e 7 colunas. A fonte original dessa base de dados é *Yeh, I. C., & Hsu, T. K. (2018). Building real estate valuation models with comparative approach through case-based reasoning. Applied Soft Computing, 65, 260-271.*

Essa base de dados fornece informações históricas acerca do mercado de avaliações imobiliárias que são coletados de Sindian Dist., New Taipei City, Taiwan. O objetivo neste trabalho está em estimar o preço das casas (na escala 10000 novo dólar tailandês por área unitária *ping* (3,3 metros quadrados)) por meio de 6 outras variáveis. A seguir, na Tabela 6.10, tem-se uma descrição das variáveis no conjunto de dados.

Tabela 6.10: Dados de preços de casas: descrição das variáveis

Variável	Nome	Descrição	Tipo
Y	house price of unit area	Preço da casa por área unitária (10000 Baht tailandês por Ping)	quantitativo
X1	transaction date	Data da transação (exemplo, 2013.250 = 2013 março)	qualitativo
X2	house age	Idade da casa(unidade: ano)	quantitativo
X3	distance to the nearest MRT station	Distância até a estação MRT mais próxima(unidade: metro)	quantitativo
X4	number of convenience stores	Número de lojas de conveniência no círculo de convivência a pé	quantitativo
X5	latitude	Latitude (unidade: grau)	quantitativo
X6	longtitude	Longitude (unidade: grau)	quantitativo

Fonte: Adaptado de *Real Estate Valuation Data Set.* Disponível em: <`https://code.datasciencedojo.com/datasciencedojo/datasets/tree/master/Real%20Estate%20Valuation`>.

Ajuste dos algoritmos

A seguir, serão apresentados os resultados dos algoritmos a partir da função objetivo J mencionada na Subseção 3.2, que faz uso do cálculo da soma dos SQE em cada *cluster*.

A partir das informações apresentadas na Tabela 6.11 é possível perceber que o algoritmo MoSCH apresentou o melhor ajuste entre todos os algoritmos, seguido

pelo algoritmo RCL, enquanto que o pior ajuste foi apresentado pelo algoritmo K-*Means* Linear. O K-*Means* híbrido apresentou o modelo SVM em 3 *clusters* e o modelo GLM em 1 *cluster*, ao passo que o algoritmo MoSCH apresentou o modelo SVM em 3 *clusters* e o modelo GAM em 1 *cluster*.

Tabela 6.11: Comparação entre os algoritmos. Base de dados de preços de casas utilizando 100 partições iniciais. Número de *cluster* = 4.

Algoritmo	Função Objetivo	Modelos
K-*Means*-Linear	23.446,8	4 GLM (gaussiana: identidade);
RCL	3.449,2	4 GLM (gaussiana: identidade);
K-*Means*-Híbrido	17.337,7	1 GLM (gaussiana: inversa) ,
		1 SVM (*kernel* radial e regressão eps),
		1 SVM (polinomial de grau 3 e regressão nu),
		1 SVM (polinomial de grau 3 e regressão eps);
MoSCH	2.018,6	1 GAM(gaussiana: identidade e bs=cr),
		3 SVM (*kernel* radial e regressão nu);

Fonte: Autoria própria

Avaliação preditiva por validação cruzada

A seguir, serão apresentados os resultados da avaliação do melhor método de alocação para o conjunto de dados desta seção. Detalhes de como foi realizada a validação cruzada podem ser encontrados na Subseção 6.1.

A Tabela 6.12 mostra os valores da média e do desvio padrão do REQM em relação aos 10 *folds* considerados na validação cruzada. Pode-se perceber que o melhor método de alocação foi a alocação com o KNN dos *clusters* combinados pois apresentou as menores médias e os menores desvios padrão, enquanto que o pior desempenho foi apresentado pelo método da alocação aleatória, com as maiores médias e os maiores desvios padrão. O melhor algoritmo foi o MoSCH, que apresentou a menor média e menor desvio padrão entre todos os algoritmos, e o algoritmo com pior desempenho foi o K-*Means* linear, apresentando a maior média e o maior desvio padrão entre os algoritmos.

Adotando um ranqueamento pela menor média do REQM, com base nas informações apresentadas na Tabela 6.12, tem-se: o MoSCH (1º), K-*Means* Híbrido (2º), RCL (3º) e K-*Means* Linear (4º). O resultado final do ranqueamento está apresentado na Seção 6.3.

Tabela 6.12: Conjunto de dados de preços de casa, comparação entre os algoritmos, por método de alocação: Média e Desvio-Padrão do REQM no conjunto Teste. Validação Cruzada 10-*fold*, utilizando 30 partições iniciais.

Algoritmo	AAle	AKNN	AKNC	Stk1	StkN
K-*Means* Linear	11,8551 (6,1360)	14,8724 (10,3175)	12,5474 (6,5921)	**11,6330** (5,0716)	11,7437 (5,7377)
RCL	16,6943 (9,6501)	**10,9686** (2,7813)	11,1650 (2,5678)	14,8731 (4,8285)	18,3299 (12,9159)
K-*Means* Híbrido	16,0216 (13,5350)	**9,6416** (3,6136)	10,5954 (4,1546)	14,9791 (10,4534)	10,2081 (4,4619)
MoSCH	15,4055 (9,9682)	9,6028 (2,4436)	**9,5393** (1,9152)	18,0114 (7,5035)	10,2839 (2,1750)

Fonte: Autoria própria

Foram realizados testes de hipóteses relacionando os métodos de alocação para cada algoritmo, os quais podem ser encontrados no Apêndice A.5. Um resumo dos resultados dos testes de hipóteses apresentados no Apêndice A pode ser encontrada na Seção 6.4. De acordo com os testes de hipótese do Apêndice A.5, todos os métodos de alocação apresentaram resultados similares para o algoritmo K-*Means* Linear, e os métodos de alocação AKNN, AKNC e StkN apresentaram resultados similares para o algoritmo MoSCH.

Também foram realizados testes de hipóteses relacionando os algoritmos, os quais podem ser encontrados no Apêndice B.5. As amostras utilizadas nos testes de hipóteses que comparam os algoritmos correspondem ao método de alocação que apresentou o menor REQM médio. De acordo com os testes de hipóteses do apêndice B.5 o melhor algoritmo foi o K-*Means* Linear.

6.2.5 Conjunto de dados do serviço de transfusão de sangue

Esse conjunto de dados possui originalmente 748 linhas (observações) e 5 colunas. A fonte dessa base de dados é o Repositório de Aprendizado de Máquina da Universidade da Califórnia, Irvine, nos EUA (UCI) e está disponível em `https://archive.ics.uci.edu/ml/index.php`.

As informações são provenientes do banco de dados de doadores do Centro de Serviço de Transfusão de Sangue na cidade de Hsin-Chu em Taiwan. Foram selecionados aleatoriamente 748 doadores registrados no banco de dados. O objetivo neste trabalho está em estimar o tempo para retorno do doador por meio de outras variáveis. A seguir, na Tabela 6.13 tem-se o dicionário dos dados.

Tabela 6.13: Dados do serviço de transfusão de sangue: descrição das variáveis relacionadas ao doador de sangue

Variável	Nome	Descrição	Tipo
Y	Recency (months)	Número de meses desde a doação mais recente	quantitativo
X1	Frequency (times)	Total de doações	quantitativo
X2	Monetary (c.c.blood)	Total de sangue doado (centímetros cúbicos)	quantitativo
X3	Time (months)	Número de meses desde a primeira doação	quantitativo
X4	whether he/shedonated blood in March 2007	Indica se doou sangue em março de 2007: (1, 0)	quantitativo

Fonte: Adaptado de *Blood Transfusion Service Center Data Set*. Disponível em: <`https://code.datasciencedojo.com/datasciencedojo/datasets/tree/master/Blood%20Transfusion%20Service%20Center`>.

Ajuste dos algoritmos

A seguir, serão apresentados os resultados dos algoritmos a partir da função objetivo J mencionada na Subseção 3.2, que faz uso do cálculo da soma dos SQE de cada *cluster*.

A partir das informações apresentadas na Tabela 6.14, é possível perceber que o algoritmo RCL apresentou o melhor ajuste entre todos os algoritmos, seguido pelo algoritmo MoSCH. O pior ajuste foi apresentadopelo algorimto K-*Means* Linear. O algoritmo K-*Means* híbrido apresentou o modelo SVM em 2 *clusters* e o modelo KNN em 3 *clusters*, enquanto que o algoritmo MoSCH apresentou o modelo KNN nos 5 *clusters*.

Tabela 6.14: Comparação entre os algoritmos. Base de dados o serviço de transfusão de sangue utilizando 100 partições iniciais. Número de *cluster* = 5.

Algoritmo	Função Objetivo	Modelos
K-*Means*-Linear	38.870,4	5 GLM (gaussiana: identidade);
RCL	836,5	5 GLM (gaussiana: identidade);
K-*Means*-Híbrido	22.368,4	1 SVM (*kernel* radial e regressão nu), 1 SVM (*kernel* radial e regressão eps), 1 KNN(k=3), 1 KNN(k=9), 1 KNN(k=11);
MoSCH	1.278,1	5 KNN(k=3);

Fonte: Autoria própria

Avaliação preditiva por validação cruzada

A seguir, serão apresentados os resultados da avaliação do melhor método de alocação para o conjunto de dados desta seção. Detalhes de como foi realizada a validação cruzada podem ser encontrados na Subseção 6.1.

A Tabela 6.15 mostra os valores da média e do desvio padrão do REQM em relação aos 10 *folds* considerados na validação cruzada. Pode-se observar que o melhor método de alocação foi a alocação com StkRegSCINN pois apresentou as menores médias e os menores desvios padrão. O pior método de alocação foi o método de alocação aleatória, que apresentou as maiores médias e os maiores desvios padrão. O melhor algoritmo foi o K-*Means* Híbrido, apresentando a menor média e menor desvio padrão entre todos os algoritmos, enquanto que o pior algoritmo foi o RCL pois apresentou a maior média e o maior desvio padrão entre os algoritmos.

Adotando um ranqueamento pela menor média do REQM, com base nas informações apresentadas na Tabela 6.15, tem-se: o K-*Means* Híbrido (1º), K-*Means* Linear (2º), MoSCH (3º) e RCL (4º). O resultado final do ranqueamento estará na Seção 6.3.

Tabela 6.15: Conjunto de dados do serviço de transfusão de sangue, comparação entre os algoritmos, por método de alocação: Média e Desvio-Padrão do REQM no conjunto Teste. Validação Cruzada 10-*fold*, utilizando 30 partições iniciais.

Algoritmo	AAle	AKNN	AKNC	Stk1	StkN
K-*Means* Linear	7,3459 (1,1844)	7,7247 (1,5727)	7,2816 (1,2083)	**7,1918** (1,2178)	7,3435 (1,3007)
RCL	11,3715 (3,8208)	10,4503 (1,3510)	**8,5422** (1,5630)	9,0545 (1,7509)	11,3584 (3,4296)
K-*Means* Híbrido	7,2948 (1,9745)	7,8535 (2,8001)	7,1593 (1,5680)	6,8373 (1,4837)	**6,3138** (1,3877)
MoSCH	9,5227 (1,3526)	9,4956 (1,0712)	7,5372 (1,3812)	8,7360 (1,7826)	**7,5260** (1,1763)

Fonte: Autoria própria

Foram realizados testes de hipóteses relacionando os métodos de alocação para cada algoritmo, os quais podem ser encontrados no Apêndice A.6. Um resumo dos resultados dos testes de hipóteses apresentados no Apêndice A pode ser encontrada na Seção 6.4. De acordo com os testes de hipótese do Apêndice A.6, os melhores métodos de alocação foram o Stk1 para o algoritmo K-*Means* Linear, o AKNC e o Stk1 para o RCL, o StkN para o algoritmo K-*Means* híbrido e o ANKC e o Stk1 para o algoritmo MoSCH.

Também foram realizados testes de hipóteses relacionando os algoritmos, os quais podem ser encontrados no Apêndice B.6. As amostras utilizadas nos testes de hipó-

teses que comparam os algoritmos correspondem ao método de alocação que apresentou o menor REQM médio. De acordo com os testes de hipóteses do apêndice B.6 os melhores algoritmos foram o K-*Means* Linear, o K-*Means* Híbrido e o MoSCH.

6.2.6 Conjunto de dados de compressão do concreto

Esse conjunto de dados possui originalmente 1030 linhas (observações) e 9 colunas. A fonte dessa base de dados é *I-Cheng Yeh, "Modeling of strength of high performance concrete using artificial neural networks", Cement and Concrete Research, Vol. 28, No. 12, pp. 1797-1808 (1998).*

As informações são provenientes de testes de concreto na engenharia civil em relação a sua resistência à compressão, a qual é uma função não linear que depende dos materiais e do tempo. O objetivo neste trabalho está em estimar a resistência à compressão do concreto por meio de outras variáveis. A seguir, na Tabela 6.16 tem-se o dicionário de dados.

Tabela 6.16: Dados de compressão do concreto: descrição das variáveis

Variável	Nome	Descrição	Tipo
Y	Concrete compressive strength (MPa,megapascals)	Resistência à compressão do concreto - MPa	quantitativo
X1	Cement (component 1) (kg in a m^3 mixture)	Cimento - kg da mistura em m^3	quantitativo
X2	Blast Furnace Slag (component 2) (kg in a m^3 mixture)	Escória de alto-forno - kg da mistura em m^3	quantitativo
X3	Fly Ash (component 3) (kg in a m^3 mixture)	Cinzas volantes - kg da mistura em m^3	quantitativo
X4	Water (component 4) (kg in a m^3 mixture)	Água - kg da mistura em m^3	quantitativo
X5	Superplasticizer (component 5) (kg in a m^3 mixture)	Superplastificante - kg da mistura em m^3	quantitativo
X6	Coarse Aggregate (component 6) (kg in a m^3 mixture)	Agregado grosso - kg da mistura em m^3	quantitativo
X7	Fine Aggregate (component 7) (kg in a m^3 mixture)	Agregado fino - kg da mistura em m^3	quantitativo
X8	Age (day)	Idade - Dia (1-365)	quantitativo

Fonte: Adaptado de *Concrete Compressive Strength Data Set.* Disponível em: <`https://code.datasciencedojo.com/datasciencedojo/datasets/tree/master/Concrete%20Compressive%20Strength`>.

Ajuste dos algoritmos

A seguir, serão apresentados os resultados dos algoritmos a partir da função objetivo J mencionada na Subseção 3.2, que faz uso do cálculo da soma dos SQE de cada *cluster*.

A partir das informações apresentadas na Tabela 6.17, é possível perceber que o algoritmo MoSCH apresentou o melhor ajuste entre todos os algoritmos, e que o algoritmo K-*Means* Linear apresentou o pior ajuste. O algoritmo RCL foi superior ao algoritmo K-*Means* Híbrido com relação ao ajuste. O algoritmo K-*Means* Híbrido apresentou o modelo SVM em 3 *clusters* e o modelo KNN em 1 textitcluster, enquanto que o algoritmo MoSCH apresentou o modelo SVM em 4 *clusters*.

Tabela 6.17: Comparação entre os algoritmos. Base de dados de compressão do concreto utilizando 100 partições iniciais. Número de *cluster* = 4.

Algoritmo	Função Objetivo	Modelos
K-*Means*-Linear	79.287,5	4 GLM (gaussiana: identidade);
RCL	8.901,4	4 GLM (gaussiana: identidade);
K-*Means*-Híbrido	25.774,8	3 SVM (*kernel* radial e regressão eps), 1 KNN (k=3);
MoSCH	6.121,9	4 SVM (*kernel* radial e regressão nu);

Fonte: Autoria própria

Avaliação preditiva por validação cruzada

A seguir, serão apresentados os resultados da avaliação do melhor método de alocação para o conjunto de dados desta seção. Detalhes de como foi realizada a validação cruzada podem ser encontrados na Subseção 6.1.

A Tabela 6.18 mostra os valores da média e do desvio padrão do REQM em relação aos 10 *folds* considerados na validação cruzada. Pode-se concluir que o melhor método de alocação foi a alocação com o KNN dos *clusters* combinados pois apresentou as menores médias e os menores desvios padrão. O método da alocação StkRegSCINN apresentou o pior resultado, com as maiores médias e os maiores desvios padrão. O melhor algoritmo foi o MoSCH, apresentando a menor média e o menor desvio padrão entre todos os algoritmos, enquanto que o pior desempenho foi apresentado pelo algoritmo RCL, com a maior média e o maior desvio padrão entre os algoritmos.

Adotando um ranqueamento pela menor média do REQM, com base nas informações apresentadas na Tabela 6.18, tem-se: o MoSCH (1º), RCL (2º), K-*Means* Híbrido (3º) e K-*Means* Linear (4º). O resultado final do ranqueamento estará na Seção 6.3.

Tabela 6.18: Conjunto de dados de compressão do concreto, comparação entre os algoritmos, por método de alocação: Média e Desvio-Padrão do REQM no conjunto Teste. Validação Cruzada 10-*fold*, utilizando 30 partições iniciais.

Algoritmo	AAle	AKNN	AKNC	Stk1	StkN
K-*Means* Linear	13,5100 (2,2192)	14,2145 (2,8548)	**12,4993** (3,2094)	14,1065 (3,9553)	16,4984 (10,7692)
RCL	17,7209 (4,8836)	19,3445 (6,8839)	**12,0958** (2,2682)	14,0036 (3,3378)	18,2120 (4,0325)
K-*Means* Híbrido	13,8287 (2,2274)	15,0089 (3,6202)	**12,2809** (1,4695)	14,3942 (4,2953)	14,3090 (2,5208)
MoSCH	11,6161 (1,1115)	10,3604 (0,9897)	**8,8710** (0,9347)	9,7616 (1,4513)	11,3938 (1,9676)

Fonte: Autoria própria

Foram realizados testes de hipóteses relacionando os métodos de alocação para cada algoritmo, os quais podem ser encontrados no Apêndice A.7. Um resumo dos resultados dos testes de hipóteses apresentados no Apêndice A pode ser encontrada na Seção 6.4. De acordo com os testes de hipótese do Apêndice A.7, os melhor métodos de alocação foi o AKCN para todos os algoritmos.

Também foram realizados testes de hipóteses relacionando os algoritmos, os quais podem ser encontrados no Apêndice B.7. As amostras utilizadas nos testes de hipóteses que comparam os algoritmos correspondem ao método de alocação que apresentou o menor REQM médio. De acordo com os testes de hipóteses do apêndice B.7 o melhor algoritmo foi o RCL.

6.2.7 Conjunto de dados de bicicletas

Esse conjunto de dados possui originalmente 731 linhas (observações) e 16 colunas, no entanto as colunas *instant*, *dteday*, *season*, *weekday*, *mnth*, *weathersit*, *casual* e *registered* não foram consideradas para a análise , algumas justificativas são que a coluna *instant* praticamente informa o índice da gravação que por sua vez seria o número da linha, a coluna *dteday* informa a data de forma textual, e a soma das colunas *casual* e *registered* seria o Y. A fonte dessa base de dados é *Fanaee-T, Hadi, and Gama, Joao, 'Event labeling combining ensemble detectors and background knowledge', Progress in Artificial Intelligence (2013): pp. 1-15, Springer Berlin Heidelberg.*

As informações são provenientes da contagem diária de bicicletas alugadas entre os anos de 2011 e 2012 no sistema Capital *bikeshare* com as informações meteorológicas e sazonais correspondentes. O objetivo neste trabalho está em estimar a contagem do total de bicicletas alugadas, incluindo casuais e registradas, por meio de

outras variáveis. Já na coluna *hum* ou X6 os valores são divididos em 100 (máx.); e também na coluna *windspeed* ou X7 os valores são divididos em 67 (máx). A seguir, na Tabela 6.19 tem-se o dicionário de dados.

Tabela 6.19: Dados de bicicletas: descrição das variáveis

Variável	Nome	Descrição	Tipo
Y	cnt	Contagem do total de bicicletas alugadas	quantitativo
X1	yr	Ano (0:2011, 1:2012)	quantitativo
X2	holiday	Indica dia de feriado (1:sim, 0:não)	quantitativo
X3	workingday	Indica dia útil (1:sim, 0:não)	quantitativo
X4	temp	Temperatura normalizada em Celsius	quantitativo
X5	atemp	Sensação de temperatura normalizada em Celsius	quantitativo
X6	hum	Umidade normalizada.	quantitativo
X7	windspeed	Velocidade normalizada do vento	quantitativo

Fonte: Adaptado de *Bike Sharing Data Set*. Disponível em: <`https://code.datasciencedojo.com/datasciencedojo/datasets/tree/master/Bike%20Sharing`>.

Ajuste dos algoritmos

A seguir, serão apresentados os resultados dos algoritmos a partir da função objetivo J mencionada na Subseção 3.2, que faz uso do cálculo da soma dos SQE de cada *cluster*.

A partir das informações apresentadas na Tabela 6.20, pode-se observar que o algoritmo MoSCH apresentou o melhor ajuste entre todos os algoritmos, seguido pelo algoritmo RCL, e que o pior ajuste foi apresentado pelo algoritmo K-*Means* Linear. O algoritmo K-*Means* híbrido apresentou o modelo SVM em 3 *clusters* e o modelo KNN em 1 *cluster*. O algoritmo MoSCH apresentou o modelo SVM em 4 *clusters*.

Tabela 6.20: Comparação entre os algoritmos. Base de dados de bicicletas utilizando 100 partições iniciais. Número de *cluster* = 4.

Algoritmo	Função Objetivo	Modelos
K-*Means*-Linear	161.852.597	4 GLM (gaussiana: identidade);
RCL	71.095.420	4 GLM (gaussiana: identidade);
K-*Means*-Híbrido	116.318.303	3 SVM (*kernel* radial e regressão eps), 1 KNN(k=11);
MoSCH	35.886.294	4 SVM (*kernel* radial e regressão nu);

Fonte: Autoria própria

Avaliação preditiva por validação cruzada

A seguir, serão apresentados os resultados da avaliação do melhor método de alocação para o conjunto de dados desta seção. Detalhes de como foi realizada a validação cruzada podem ser encontrados na Subseção 6.1.

A Tabela 6.21 apresenta os valores da média e do desvio padrão do REQM em relação aos 10 *folds* considerados na validação cruzada. Pode-se observar que o melhor método de alocação foi a alocação com o KNN dos *clusters* combinados pois apresentou as menores médias e os menores desvios padrão. O pior desempenho foi apresentado pelo método da alocação StkRegSCI1, com as maiores médias e os maiores desvios padrão. O melhor algoritmo foi o MoSCH, pois apresentou a menor média e menor desvio padrão entre todos os algoritmos, enquanto que o pior algoritmo foi o K-*Means* Híbrido pois apresentou a maior média e o maior desvio padrão entre quase todos algoritmos.

Adotando um ranqueamento pela menor média do REQM, com base nas informações apresentadas na Tabela 6.21, tem-se: o MoSCH (1º), RCL (2º), K-*Means* Híbrido (3º) e K-*Means* Linear (4º). O resultado final do ranqueamento estará na Seção 6.3.

Tabela 6.21: Conjunto de dados de bicicletas, comparação entre os algoritmos, por método de alocação: Média e Desvio-Padrão do REQM no conjunto Teste. Validação Cruzada 10-*fold*, utilizando 30 partições iniciais.

Algoritmo	AAle	AKNN	AKNC	Stk1	StkN
K-*Means* Linear	2.156,53 (860,27)	2.279,84 (897,72)	2.124,69 (664,94)	**1.853,19** (682,91)	1.917,72 (705,53)
RCL	1.516,20 (294,46)	1.262,99 (290,72)	**1.143,40** (117,96)	1.561,71 (392,99)	1.637,06 (396,56)
K-*Means* Híbrido	2.274,53 (955,04)	2.334,32 (939,46)	2.815,46 (1.056,07)	3.309.402 (10.457.012)	**1.808,10** (545,08)
MoSCH	1.064,99 (101,33)	1.051,40 (94,87)	**884,86** (63,80)	1.122,69 (108,65)	1.131,70 (99,25)

Fonte: Autoria própria

Foram realizados testes de hipóteses relacionando os métodos de alocação para cada algoritmo, os quais podem ser encontrados no Apêndice A.8. Um resumo dos resultados dos testes de hipóteses apresentados no Apêndice A pode ser encontrada na Seção 6.4. De acordo com os testes de hipótese do Apêndice A.8, os melhores métodos de alocação foram o AKNC para os algoritmos K-*Means* Híbrido, RCL e MoSCH, e o AAle para o algoritmo K-*Means* Linear.

Também foram realizados testes de hipóteses relacionando os algoritmos, os quais podem ser encontrados no Apêndice B.8. As amostras utilizadas nos testes de hipóteses que comparam os algoritmos correspondem ao método de alocação que apresentou o menor REQM médio. De acordo com os testes de hipóteses do apêndice B.8 o melhor algoritmo foi o K-*Means* Híbrido.

6.2.8 Conjunto de dados de abalone

Esse conjunto de dados possui originalmente 4177 linhas (observações) e 9 colunas, no entanto foram ignoradas as colunas *Viscera weight* e *Shell weight* pois a soma das duas resultará no próprio Y, e também para descobrir o peso das vísceras e da concha seria necessário matar o animal. A fonte dessa base de dados é *Warwick J Nash, Tracy L Sellers, Simon R Talbot, Andrew J Cawthorn and Wes B Ford (1994) "The Population Biology of Abalone (Haliotis species) in Tasmania. I. Blacklip Abalone (H. rubra) from the North Coast and Islands of Bass Strait", Sea Fisheries Division, Technical Report No. 48 (ISSN 1034-3288).*

Essa base de dados é proveniente de medições físicas de abalones e o número de anéis (representando a idade). A idade do abalone pode ser determinada cortando a concha através do cone, colorindo-a e contando o número de anéis através de um microscópio, uma tarefa tediosa e demorada. Como o abalone é um molusco

apreciado na gastronomia e criado em fazendas de criação, o objetivo neste trabalho está em estimar o quanto de carne um abalone pode fornecer com base em outras variáveis. A variável *Sex* ou X1 originalmente tem seus valores no conjunto (M: Masculino, F: Feminino, I: Infantil). A seguir na Tabela 6.22 tem-se o dicionário de dados.

Tabela 6.22: Dados de abalone: descrição das variáveis

Variável	Nome	Descrição	Tipo
Y	Shucked weigh	Peso da carne (g)	quantitativo
X1	Sex	Sexo (1: Feminino, 2: Infantil, 3: Masculino)	qualitativo
X2	Length	Medida mais longa da casca (mm)	quantitativo
X3	Diameter	Diâmetro - perpendicular ao comprimento (mm)	quantitativo
X4	Height	Altura - com carne na casca (mm)	quantitativo
X5	Whole weight	Peso do abalone inteiro (g)	quantitativo
X6	Rings	Anéis - valor +1,5 dá a idade em anos (por exemplo, 4 = 5,5 anos)	quantitativo

Fonte: Adaptado de *Abalone Data Set*. Disponível em: <`https://code.datasciencedojo.com/datasciencedojo/datasets/tree/master/Abalone`>.

Ajuste dos algoritmos

A seguir, serão apresentados os resultados dos algoritmos a partir da função objetivo J mencionada na Subseção 3.2, que faz uso do cálculo da soma dos SQE de cada *cluster*.

A partir das informações apresentadas na Tabela 6.23, pode-se perceber que o algoritmo RCL apresentou o melhor ajuste entre todos os algoritmos, seguido pelo algoritmo MoSCH, enquanto que o pior ajuste foi apresentado pelo algoritmo K-*Means* Linear. O algoritmo K-*Means* híbrido apresentou o modelo SVM em 2 *clusters*, o modelo Ctree em 1 *cluster* e o modelo KNN em 1 *cluster*. O algoritmo MoSCH apresentou o modelo SVM em 2 *clusters*, o modelo Ctree em 1 *cluster* e o modelo KNN em 1 *cluster*.

Tabela 6.23: Comparação entre os algoritmos. Base de dados de abalone utilizando 100 partições iniciais. Número de *cluster* = 4.

Algoritmo	Função Objetivo	Modelos
K-*Means*-Linear	8,9647	4 GLM (gaussiana: identidade);
RCL	1,4671	4 GLM (gaussiana: identidade);
K-*Means*-Híbrido	7,3354	2 SVM (*kernel* radial e regressão nu), 1 KNN (k=3), 1 Ctree;
MoSCH	1,4971	2 SVM (*kernel* radial e regressão nu), 1 KNN (k=3), 1 Ctree;

Fonte: Autoria própria

Avaliação preditiva por validação cruzada

A seguir, serão apresentados os resultados da avaliação do melhor método de alocação para o conjunto de dados desta seção. Detalhes de como foi realizada a validação cruzada podem ser encontrados na Subseção 6.1.

A Tabela 6.24 apresenta os valores da média e do desvio padrão do REQM em relação aos 10 *folds* considerados na validação cruzada. Pode-se observar que o melhor método de alocação foi a alocação com o KNN dos *clusters* combinados, apresentando as menores médias e os menores desvios-padrão. O pior desempenho foi apresentado pelo método de alocação aleatória, com as maiores médias e os maiores desvios padrão. O melhor algoritmo foi o K-*Means* Linear que apresentou as menores médias e os menores desvios-padrão entre todos os algoritmos. O pior desempenho foi apresentado pelo algoritmo K-*Means* Híbrido, que apresentou a maior média e o maior desvio-padrão entre os algoritmos.

Adotando um ranqueamento com base na menor média do REQM, com base nas informações apresentadas na Tabela 6.24, tem-se: o K-*Means* Linear (1º), RCL (2º), MoSCH (3º) e K-*Means* Híbrido (4º). O resultado final do ranqueamento estará na Seção 6.3.

Tabela 6.24: Conjunto de dados de abalone, comparação entre os algoritmos, por método de alocação: Média e Desvio-Padrão do REQM no conjunto Teste. Validação Cruzada 10-*fold*, utilizando 30 partições iniciais.

Algoritmo	AAle	AKNN	AKNC	Stk1	StkN
K-*Means* Linear	0,0522 (0,0054)	0,0508 (0,0049)	**0,0477** (0,0031)	0,0492 (0,0029)	0,0519 (0,0040)
RCL	0,0875 (0,0447)	0,0502 (0,0056)	**0,0496** (0,0046)	0,0733 (0,0242)	0,0892 (0,0437)
K-*Means* Híbrido	0,0925 (0,0552)	0,0897 (0,0565)	**0,0693** (0,0289)	0,1015 (0,0748)	0,0782 (0,0389)
MoSCH	0,0836 (0,0476)	0,0687 (0,0231)	**0,0590** (0,0066)	0,1476 (0,1127)	0,0836 (0,0172)

Fonte: Autoria própria

Foram realizados testes de hipóteses relacionando os métodos de alocação para cada algoritmo, os quais podem ser encontrados no Apêndice A.9. Um resumo dos resultados dos testes de hipóteses apresentados no Apêndice A pode ser encontrada na Seção 6.4. De acordo com os testes de hipótese do Apêndice A.9, os melhores métodos de alocação foram o AKNC para os algoritmos K-*Means* Linear, K-*Means* Híbrido e MoSCH, e o AKNN e AKNC para o algoritmo RCL.

Também foram realizados testes de hipóteses relacionando os algoritmos, os quais podem ser encontrados no Apêndice B.9. As amostras utilizadas nos testes de hipóteses que comparam os algoritmos correspondem ao método de alocação que apresentou o menor REQM médio. De acordo com os testes de hipóteses do apêndice B.9 o melhor algoritmo foi o MoSCH.

6.3 O algoritmo vencedor em relação as bases de dados

A seguir, a Tabela 6.25 apresenta o ranqueamento dos algoritmos apresentados na Seção 6.2. Foi adotado um ranqueamento pela menor média do REQM do algoritmo da linha (a menor média do algoritmo entre os métodos de alocação) referente a respectiva tabela com os resultados da validação cruzada. Pode-se concluir que o melhor desempenho foi apresentado pelo algoritmo MoSCH, que apresentou a menor soma das posições do ranqueamento e o pior desempenho foi apresentado pelos algoritmos K-*Means* Linear e RCL, que apresentaram as maiores somas das posições do ranqueamento.

Tabela 6.25: Ranqueamentos das bases de dados (BD) da Seção 6.2

Algoritmo	BD 6.3.1	BD 6.3.2	BD 6.3.3	BD 6.3.4	BD 6.3.5	BD 6.3.6	BD 6.3.7	BD 6.3.8	Soma
K-*Means* Linear	3	4	1	4	2	4	4	1	23
RCL	4	2	4	3	4	2	2	2	23
K-*Means* Híbrido	1	3	3	2	1	3	3	4	20
MoSCH	2	1	2	1	3	1	1	3	14

Fonte: Autoria própria

6.4 Os métodos de alocação vencedores em relação as bases de dados

A seguir, a Tabela 6.26 apresenta os métodos de alocação vencedores agrupados pelos algoritmos considerados na Seção 6.2 com base nas tabelas obtidas a partir da validação cruzada. Foram realizados testes de hipóteses, os quais podem ser visualizados no Apêndice A. Utilizando o critério do voto majoritário, o método de alocação que apresentou o melhor desempenho foi o AKNC, enquanto que o método de alocação que apresentou o pior desempenho foi o AAle, que somente figurou entre os melhores algoritmos quando a diferença entre os parâmetros de locação não foi significativa, de acordo com os testes de hipóteses.

Tabela 6.26: Compilado dos testes de hipóteses no Apêndice A das bases de dados (BD) da Seção 6.2 para os métodos de alocação.

Algoritmo	BD 6.3.1	BD 6.3.2	BD 6.3.3	BD 6.3.4	BD 6.3.5	BD 6.3.6	BD 6.3.7	BD 6.3.8	Eleito
K-*Means* Linear	AKNC	AKNC	AKNC	Todos	Stk1	AKNC	AAle	AKNC	AKNC
RCL	Stk1	AKNC	AKNC	AKNN	AKNC Stk1	AKNC	AKNC	AKNN AKNC	AKNC
K-*Means* Híbrido	AKNC	StkN	AKNC AKNN	AKNN AKNC StkN	StkN	AKNC	AKNC	AKNC	AKNC
MoSCH	AKNC StkN	StkN	StkN	AKNN AKNC StkN	AKNC StkN	AKNC	AKNC Stk1	AKNC	AKNC

Fonte: Autoria própria

Capítulo 7

Trabalhos futuros

Como trabalhos futuros estão a utilização de outras fórmulas para serem usadas como método de alocação fazendo uso dos vizinhos próximos. A simbologia utilizada pode ser conferida na Seção 4.2.

A média ponderada com inverso multiplicativo da distância pode ser estudada, mas o problema é que se qualquer distância for 0 faz com que haja indeterminação. A fórmula da média ponderada com inverso multiplicativo da distância pode ser descrita como:

$$MPI = \begin{cases} \frac{\sum_{i \in I} M_{G_i \in PP}(r)/D_i}{\sum_{i \in I} 1/D_i} & \text{se } \prod_{i \in I} D_i \neq 0 \\ \sum_{i \in I} M_{G_i \in PP}(r)/k & \text{se } \prod_{i \in I} D_i = 0 \end{cases}$$

Outra média que aparentemente poderia ser útil é a média geométrica ponderada. Computacionalmente devido as sucessivas multiplicações deve-se trabalhar com *Big Decimal* pois um número *double* pode ser facilmente estourado. Um fato é que nada impede que uma predição possa ser um número negativo, fazendo com que o conteúdo da raiz seja negativo levando o problema para o conjunto dos números complexos, por via de regra a média geométrica não admite números negativos. Devido a isso para não restringir a predição a apenas valores positivos e por ser mais usadas em problemas que tem aumentos sucessivos e taxas de crescimento pode ser um problema trabalhar com ela.

Uma possível ideia para fazer com que a média geométrica trabalhe com números negativos é colocar uma compensação, mas não foi analisado se isso é possível, pode ser que seja inviável. Deve-se avaliar se pode ser posto essa compensação. C é a compensação e tem valor 0 se todos os $M_{G_i \in P}(r)$ com $i \in I$ são positivos, este é o caso convencional. Caso contrário C tem o valor menor que o menor $M_{G_i \in P}(r)$ com $i \in I$, achar um bom valor para C é também um problema. A fórmula da média

geométrica ponderada com compensação pode ser vista como:

$$MG = \begin{cases} \sqrt[\sum_{i \in I}(S-D_i)]{\prod_{i \in I}(M_{G_i \in PP}(r) - C)^{(S-D_i)}} + C & \text{se } S \neq 0 \\ \sum_{i \in I} M_{G_i \in PP}(r)/k & \text{se } S = 0 \end{cases}$$

A média harmônica ponderada utilizando o inverso multiplicativo da distância como peso tem novamente o problema da divisão por 0 que pode ocorrer com mais freiuência, tanto pelo motivo de alguma distância do vizinho próximo ser 0 como também pela predição do *cluster* ser 0. A fórmula é:

$$MHI = \begin{cases} \frac{\sum_{i \in I} 1/D_i}{\sum_{i \in I} 1/(D_i * M_{G_i \in PP}(r))} & \text{se } \prod_{i \in I} D_i \neq 0 \text{ e } \prod_{i \in I} M_{G_i \in PP}(r) \neq 0 \\ \frac{k}{\sum_{i \in I} 1/(M_{G_i \in PP}(r))} & \text{se } \prod_{i \in I} D_i = 0 \text{ e } \prod_{i \in I} M_{G_i \in PP}(r) \neq 0 \\ \frac{\sum_{i \in I} M_{G_i \in PP}(r)}{k} & \text{se } \prod_{i \in I} M_{G_i \in PP}(r) = 0 \end{cases}$$

Uma forma alternativa de se escrever a média harmônica ponderada é utilizar o peso como a diferença entre a soma das distâncias de todos os vizinhos e a distância daiuele vizinho próximo. Esse caso evitaria mais a indeterminação com divisão por 0. A formula pode ser vista como:

$$MHW = \begin{cases} \frac{\sum_{i \in I}(S-Di)}{\sum_{i \in I}(S-Di)/(M_{G_i \in PP}(r))} & \text{se } S \neq 0 \text{ e } \prod_{i \in I} M_{G_i \in PP}(r) \neq 0 \\ \frac{k}{\sum_{i \in I} 1/(M_{G_i \in PP}(r))} & \text{se } S = 0 \text{ e } \prod_{i \in I} M_{G_i \in PP}(r) \neq 0 \\ \frac{\sum_{i \in I} M_{G_i \in PP}(r)}{k} & \text{se } \prod_{i \in I} M_{G_i \in PP}(r) = 0 \end{cases}$$

Outra fórmula bem mais simples e que foi usada em conjunto com o algoritmo KNN dos *clusters* combinados, mas não leva em consideração a distância que os vizinhos estão entre si, seria o uso da média aritmética das predições por meio dos *cluster* para as novas observações. quando as distâncias entre os vizinhos são muito próxima ou quando a distância entre os vizinhos não é significante pode valer a pena apenas o seu uso, nesse sentido a fórmula seria:

$$MA = \sum_{i \in I} M_{G_i \in PP}(r)/k$$

Também pode ser possível a utilização da Matriz de Projeção informada na Subseção 2.1.2 com a ideia de alocação de novas observações:

1. n é quantidade de observações;
2. i é o índice da observação;
3. PP é um vetor de índices de todos os *clusters*, com $\#PP$ sendo a sua cardinalidade;

4. H é a Matriz de Projeção;

5. G_i é o *cluster* de X_i, sendo G_i um índice diferente de vazio;

6. $M_{G_i \in PP}(r)$ é a predição do modelo do *cluster* de X_i para uma nova observação r;

7. c é um vetor de tamanho n em que o $c_i = M_{G_i \in PP}(r)$;

8. z é um vetor de tamanho n que informará o quão perto o valor da nova observação r está em relação as observações já existentes. Assim $z = c - Hc = (I - H)c$, e quanto mais próximo de zero for o z_i melhor será.

9. A função indicadora $\delta_{G_i=k}$ é igual a 1 se o objeto i tem o *cluster* de índice $k \in PP$ e é 0 caso contrário;

10. Por último é necessário definir uma métrica em relação a z para se estimar o valor da alocação com base nos *clusters*, a situação mais comum é utilizar a predição $M_{G_i \in PP}(r)$ do *cluster* $k \in PP$ em que $\sum_{i=1}^{n} z_i * \delta_{G_i=k}$ é o menor, com $k = 1, 2, ..., \#PP$ testando todos o índices de PP.

Capítulo 8

Considerações Finais

Este trabalho é nomeado como Modelo de Segmentação *Clusterwise* com Protótipos Híbridos, em que o termo Segmentação *Clusterwise* é devido que à abordagem que segmenta os dados em *clusters* e cada *cluster* vai tratar os seus dados de acordo com o melhor protótipo híbrido ou modelo que pode ser associado.

O objetivo do MoSCH é encontrar e ajustar o melhor modelo para cada *cluster* partindo de algumas partições iniciais, de modo a minimizar a função objetivo, sendo que a partição dentro do algoritmo é mudada a cada iteração por conta da afetação.

Toda a arguição desde a fundamentação teórica até os resultados foram para dar entendimento, expor e validar três pontos essenciais. O primeiro é validade do MoSCH; o segundo é uma ideia de como pode ser feita a escolha da quantidade de *clusters* para a comparação entre métodos baseados no K-Means; e o terceiro seria mostrar o novo método de alocação baseado no KNN o qual é chamado nesse trabalho de alocação com KNN dos *Clusters* Combinados, tendo o mnemônico AKNC.

Nos resultados obtidos nos experimentos com dados sintéticos teve a comparação entre dois algoritmos e o melhor foi o MoSCH, em todos os cenários lá desenvolvidos, tanto em relação ao REQM médio como em relação aos testes de hipóteses.

Os resultados obtidos nos experimentos com dados reais para a alocação com KNN dos *Clusters* Combinados foram bem satisfatória pois se destacou entre os outros 5 métodos de alocação, sendo o melhor avaliado pelo voto majoritário para os testes de hipótese. Já em relação ao Algoritmo MoSCH nos experimentos com dados reais ele foi o melhor ranqueado entre os outros 4 algoritmos em relação ao REQM médio.

Algumas das desvantagens do algoritmo MoSCH ~~estar~~estão em depender das partições iniciais e do desempenho dos protótipos híbridos ou modelos colocados para o problema. No entanto em várias situações ele se mostrou interessante, se adequando muito bem aos dados, apesar que uma melhor adequação (ou ajuste do algoritmo) nos dados não significa necessariamente um maior acerto. O MoSCH é um algoritmo bastante robusto e escalável principalmente em relação aos modelos

que podem ser usados como protótipos híbridos, podendo ser usado vários modelos de naturezas diversas como linear, não-linear e não paramétricos. Como trabalho futuro foram apresentados novos possíveis métodos de alocação baseados no KNN, os quais podem ser desenvolvidos ou descartados. A questão do problema das partições iniciais também é algo que pode ser melhorado com alguma abordagem utilizando algoritmo genético, ou algum outro algoritmo evolutivo.

Agradecimentos por fomento a pesquisa acadêmica

Este trabalho foi subsidiado por meio de bolsa acadêmica pela Fundação de Apoio à Pesquisa do Estado da Paraíba (FAPESQ-PB), tendo como tema Internet das Coisas. A Internet das Coisas é um conceito que se refere à interconexão digital de máquinas, sistemas ou dispositivos com a internet, sendo que a conexão tem pouca ou nenhuma interferência humana. O tema Internet das coisas é bastante amplo, e este trabalho por sua vez se relacionar com esse tema na questão de análise de dados, devido a robustez do modelo MoSCH o uso se daria no pós-processamento das coletas de informações de sensores, podendo verificar ou obter algum tipo de relação entre as variáveis medidas. No entanto este trabalho não se limita somente a Internet das Coisas, podendo ter uso em qualquer tipo de análise de dados que possa envolver o tema *Clusterwise*. Assim sendo agradece-se a FAPESQ-PB pelo incentivo ao trabalho aqui exposto, a Universidade Federal da Paraíba (UFPB) pela estrutura fornecida e aos orientadores pelo tempo dedicado a correção, explicação e orientação visando o entendimento e o aprendizado do orientando.

Referências Bibliográficas

[1] ANDREANI, R., 2009. "Revisão". Disponível em: <"http://www.ime.unicamp.br/~andreani/matrizes/lista0">.

[2] BROCK, G., PIHUR, V., DATTA, S., et al., 2011. "clValid, an R package for cluster validation". Disponível em: <https://cran.r-project.org/web/packages/clValid/vignettes/clValid.pdf>.

[3] CAMARGO, H., 2007. "Agrupamento fuzzy". Disponível em: <http://www2.dc.ufscar.br/~heloisa/SN2007/AgrupamentoFuzzy.pdf>.

[4] CARVALHO, F., CARVALHO, A., SAPORTA, G., et al., 2010, "A Clusterwise Center and Range Regression Model for Interval-Valued Data", *from book Proceedings of COMPSTAT'2010. 19th international conference on computational statistics, Paris, France, August 22–27, 2010. Keynote, invited and contributed papers*, (08). doi: 10.1007/978-3-7908-2604-3_45.

[5] CARVALHO, F., LIMA NETO, E., SILVA, K., 2020, "A clusterwise nonlinear regression algorithm for interval-valued data", *Information Sciences*. ISSN: 0020-0255. doi: https://doi.org/10.1016/j.ins.2020.10.054. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0020025520310379>.

[6] CHRISTOPHER MANNING, PRABHAKAR RAGHAVAN, H. S., 2008, *Introduction to Information Retrieval*. University of Wisconsin-Madison. Dean W. Wichern, Texas A&M UniversityCambridge University Press, Cambridge University Press. ISBN: 0521865719, 978-0521865715. Disponível em: <https://nlp.stanford.edu/IR-book/html/htmledition/centroid-clustering-1.html>.

[7] CRAN, 2019. "FAQ - CRAN". Disponível em: <https://cran.r-project.org/doc/FAQ/R-FAQ.html#What-is-CRAN_003f>.

[8] CUNHA, L., 2017. "ANÁLISE DE AGRUPAMENTO". Disponível em: <https://edisciplinas.usp.br/pluginfile.php/3498333/mod_resource/content/0/AULA4-2017.pdf>.

[9] EVERITT, B. S., LANDAU, S., LEESE, M., 2009, *Cluster Analysis*. 4th ed. King's College London, UK, Wiley Publishing. ISBN: 0470689358, 9780470689356.

[10] GALIMBERTI, G., SOFFRITTI, G., 2019, "Seemingly unrelated clusterwise linear regression", *Advances in Data Analysis and Classification*, (08). doi: 10.1007/s11634-019-00369-4.

[11] GAUSS CORDEIRO, E. L., 2013, "MODELOS PARAMETRICOS". In: *MODELOS PARAMETRICOS*, cap. 1, "Universidade Federal Rural de Pernambuco, Rua Dom Manoel de Medeiros, s/n, 50.171-900 – Recife, PE, Brasil", UFJF. Disponível em: <"http://www.ufjf.br/clecio_ferreira/files/2013/05/Livro-Gauss-e-Eufrasio.pdf">.

[12] GLOVER, F., 2016. "Pseudo-Centroid Clustering". Disponível em: <https://arxiv.org/ftp/arxiv/papers/1607/1607.03467.pdf>.

[13] GOWER, J., K., S., et al., 1985, "measures of similarity dissimilarity and distance". In: *Encyclopedia of Statistical Sciences*, Wiley. Disponível em: <https://books.google.com.br/books?id=kopFAAAAYAAJ>.

[14] GOWER, J., LEGENDRE, et al., 1986, "Metric and Euclidean properties of dissimilarity coefficients", *Journal of Classification*, v. 3, n. 1, pp. 5–48. Disponível em: <https://EconPapers.repec.org/RePEc:spr:jclass:v:3:y:1986:i:1:p:5-48>.

[15] GOWER, J. C., WARRENS, M. J., 2017, "Similarity, Dissimilarity, and Distance, Measures of". In: *Wiley StatsRef*, pp. 1–11, Wiley, 5. doi: 10.1002/9781118445112.stat02470.pub2.

[16] JOHNSON, R., 1992, *Applied Multivariate Statistical Analysis*. 3th ed. University of Wisconsin-Madison. Dean W. Wichern, Texas A&M University, Pearson. ISBN: 9780130417732.

[17] JOSHI, A., 2019, "Regression". In: *Machine Learning and Artificial Intelligence*, pp. 33–35, Microsoft (United States), Redmond, WA, USA, Springer, 09. ISBN: 978-3-030-26621-9. doi: 10.1007/978-3-030-26622-6_19.

[18] KAUFMAN L., R. P. J., 1990, "Finding Groups in Data - An Introduction to Cluster Analysis". In: *Finding Groups in Data - An Introduction to Cluster Analysis*, New York, John Wiley & Sons. ISBN: 978-0-471-87876-6. doi: 10.1002/9780470316801.

[19] KAUFMANN, L., ROUSSEEUW, P., 1987, "Clustering by Means of Medoids", *Data Analysis based on the L1-Norm and Related Methods*, (01), pp. 405–416.

[20] LANCE, G. N., WILLIAMS, W. T., 1966, "Computer Programs for Hierarchical Polythetic Classification ("Similarity Analyses")", *The Computer Journal*, v. 9, n. 1 (05), pp. 60–64. ISSN: 0010-4620. doi: 10.1093/comjnl/9.1.60. Disponível em: <https://doi.org/10.1093/comjnl/9.1.60>.

[21] LIMA NETO, E. D. A., CARVALHO, F. D. A. D., 2017, "Nonlinear regression applied to interval-valued data", *Pattern Analysis and Applications*, v. 20, n. 3, pp. 809–824. doi: 10.1007/s10044-016-0538-y.

[22] LOUREIRO, J., 2005. "Técnicas de agrupamento de dados na mineração de dados químicos". Disponível em: <https://repositorio.ufpe.br/handle/123456789/2791>.

[23] MACQUEEN, J., 1967, "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pp. 281–297, Berkeley, Calif. University of California Press. Disponível em: <https://projecteuclid.org/euclid.bsmsp/1200512992>.

[24] MONARD, M. C., BARANAUSKAS, J. A., 2003, "Indução de Regras e Árvores de Decisão". In: *Sistemas Inteligentes - Fundamentos e Aplicações*, 1 ed., Manole Ltda, pp. 39–56, Barueri-SP. ISBN: 85-204-168.

[25] PIERSON, L., 2019, *Data Science Para Leigos: Tradução da 2a Edição*. Para Leigos. Rua Viúva Cláudio, 291 - Bairro Industrial do Jacaré, CEP: 20.970-031 - Rio de Janeiro - RJ, Alta Books. ISBN: 9788550808710. Disponível em: <https://books.google.com.br/books?id=eMCODwAAQBAJ>.

[26] R. O. DUDA, P. E. HART, D. G. S., 2001, "Pattern Classification, 2 edn". In: *Pattern Classification, 2 edn*, New York, John Wiley & Sons, . ISBN: 978-0-471-05669-0.

[27] R. O. DUDA, P. E. HART, D. G. S., 2001, "Introduction to Statistical Pattern Recognition, 2 edn". In: *Introduction to Statistical Pattern Recognition, 2 edn*, San Diego, Academic, . ISBN: 978-0-080-47865-4.

[28] R-PROJECT, 2019. "What is R?" Disponível em: <https://www.r-project.org/about.html>.

[29] RENCHER, A. C., 2002, "Multivariate Regression". In: *Multivariate Analysis, 2nd edition*, pp. 322–343, Brigham Young University, USA, Wiley-Interscience. ISBN: 0-471-41889-7.

[30] RIPLEY, B. D., 1996, "Pattern Recognition and Neural Networks". In: *Pattern Recognition and Neural Networks*, Cambridge, Cambridge University Press. ISBN: 978-0521717700.

[31] RSTUDIO, 2019. "About RStudio". Disponível em: <https://rstudio.com/about/>.

[32] SAITTA, S., RAPHAEL, B., SMITH, I. F. C., 2007, "A Bounded Index for Cluster Validity". In: *MLDM*. Disponível em: <https://www.semanticscholar.org/paper/A-Bounded-Index-for-Cluster-Validity-Saitta-Raphael/2fb5b1707e5ebec72ff8a0c8e75dd0fd00256d8d>.

[33] SARAJANE PERES, C. L., 2016. "Medidas de avalia¸c˜ao de agrupamentos (Clustering)". Disponível em: <http://each.uspnet.usp.br/sarajane/wp-content/uploads/2015/11/avaliacao_clustering.pdf>.

[34] TREVINO, A., 2016. "Introduction to K-means Clustering". Disponível em: <https://blogs.oracle.com/datascience/introduction-to-k-means-clustering>.

Apêndice A

Tabelas dos testes de hipóteses dos métodos de alocação

A.1 Teste de hipótese: legenda para os símbolos

Tabela A.1: Legenda para os testes de hipótese

Símbolo	Descrição
$-$	Comparar que o método da linha é menor que o método da coluna teve um *p-valor* (nível descritivo ou probabilidade de significância) menor que o caso contrário ($+$) e o *p-valor* foi menor que a significância.
$+$	Comparar que o método da linha é maior que o método da coluna teve um *p-valor* menor que o caso contrário ($-$) e o *p-valor* foi menor que a significância.
$=$	O menor *p-valor* das comparações ($+$ ou $-$) foi maior ou igual que a significância.

Fonte: Autoria própria

A.2 Teste de hipótese: conjunto de dados de eficiência energética

Tabela A.2: Conjunto de dados de eficiência energética, teste de hipótese com significância de 10% para o algoritmo K-*Means* Linear.

Algoritmo K-*Means* Linear	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	+	=
		AKNN	+	+	=
			AKNC	−	−
				Stk1	−
					StkN

Fonte: Autoria própria

Tabela A.3: Conjunto de dados de eficiência energética, teste de hipótese com significância de 10% para o algoritmo RCL.

Algoritmo RCL	AAle	AKNN	AKNC	Stk1	StkN
	AAle	+	+	+	=
		AKNN	+	+	−
			AKNC	+	−
				Stk1	−
					StkN

Fonte: Autoria própria

Tabela A.4: Conjunto de dados de eficiência energética, teste de hipótese com significância de 10% para o algoritmo K-*Means* Híbrido.

Algoritmo K-*Means* Híbrido	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	=	+
		AKNN	+	=	+
			AKNC	−	−
				Stk1	+
					StkN

Fonte: Autoria própria

Tabela A.5: Conjunto de dados de eficiência energética, teste de hipótese com significância de 10% para o algoritmo MeSCH.

Algoritmo MeSCH	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	=	+
		AKNN	+	=	+
			AKNC	−	=
				Stk1	+
					StkN

Fonte: Autoria própria

A.3 Teste de hipótese: conjunto de dados de conjunto de dados auto MPG

Tabela A.6: Conjunto de dados de conjunto de dados auto MPG, teste de hipótese com significância de 10% para o algoritmo K-*Means* Linear.

Algoritmo K-*Means* Linear	AAle	AKNN	AKNC	Stk1	StkN
AAle		=	+	=	=
AKNN			+	=	=
AKNC				−	−
Stk1					=
StkN					

Fonte: Autoria própria

Tabela A.7: Conjunto de dados de conjunto de dados auto MPG, teste de hipótese com significância de 10% para o algoritmo RCL.

Algoritmo RCL	AAle	AKNN	AKNC	Stk1	StkN
AAle		+	+	=	+
AKNN			+	=	−
AKNC				−	−
Stk1					=
StkN					

Fonte: Autoria própria

Tabela A.8: Conjunto de dados de conjunto de dados auto MPG, teste de hipótese com significância de 10% para o algoritmo K-*Means* Híbrido.

Algoritmo K-*Means* Híbrido	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	=	−	+
		AKNN	=	−	=
			AKNC	−	=
				Stk1	+
					StkN

Fonte: Autoria própria

Tabela A.9: Conjunto de dados de conjunto de dados auto MPG, teste de hipótese com significância de 10% para o algoritmo MeSCH.

Algoritmo MeSCH	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	−	+
		AKNN	=	−	+
			AKNC	−	=
				Stk1	+
					StkN

Fonte: Autoria própria

A.4 Teste de hipótese: conjunto de dados de consumo de álcool

Tabela A.10: Conjunto de dados de consumo de álcool, teste de hipótese com significância de 10% para o algoritmo K-*Means* Linear.

Algoritmo K-*Means* Linear	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	=	=
		AKNN	+	=	−
			AKNC	−	−
				Stk1	=
					StkN

Fonte: Autoria própria

Tabela A.11: Conjunto de dados de consumo de álcool, teste de hipótese com significância de 10% para o algoritmo RCL.

Algoritmo RCL	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	=	=
		AKNN	+	−	=
			AKNC	−	=
				Stk1	=
					StkN

Fonte: Autoria própria

Tabela A.12: Conjunto de dados de consumo de álcool, teste de hipótese com significância de 10% para o algoritmo K-*Means* Híbrido.

Algoritmo K-*Means* Híbrido	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	=	−	−
		AKNN	=	=	−
			AKNC	−	=
				Stk1	=
					StkN

Fonte: Autoria própria

Tabela A.13: Conjunto de dados de consumo de álcool, teste de hipótese com significância de 10% para o algoritmo MeSCH.

Algoritmo MeSCH	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	=	=	+
		AKNN	=	=	+
			AKNC	+	+
				Stk1	+
					StkN

Fonte: Autoria própria

A.5 Teste de hipótese: conjunto de dados de preços de casa

Tabela A.14: Conjunto de dados de preços de casa, teste de hipótese com significância de 10% para o algoritmo K-*Means* Linear.

Algoritmo K-*Means* Linear	AAle	AKNN	AKNC	Stk1	StkN
AAle	=	=	=	=	
AKNN	=	=	=		
AKNC	=	=			
Stk1	=				
StkN					

Fonte: Autoria própria

Tabela A.15: Conjunto de dados de preços de casa, teste de hipótese com significância de 10% para o algoritmo RCL.

Algoritmo RCL	AAle	AKNN	AKNC	Stk1	StkN
AAle	+	=	=	=	
AKNN	=	−	−		
AKNC	−	−			
Stk1	=				
StkN					

Fonte: Autoria própria

Tabela A.16: Conjunto de dados de preços de casa, teste de hipótese com significância de 10% para o algoritmo K-*Means* Híbrido.

Algoritmo K-*Means* Híbrido	AAle	AKNN	AKNC	Stk1	StkN
	AAle	+	+	=	+
		AKNN	=	−	=
			AKNC	−	=
				Stk1	+
					StkN

Fonte: Autoria própria

Tabela A.17: Conjunto de dados de preços de casa, teste de hipótese com significância de 10% para o algoritmo MeSCH.

Algoritmo MeSCH	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	=	=	=
		AKNN	=	−	=
			AKNC	−	=
				Stk1	+
					StkN

Fonte: Autoria própria

A.6 Teste de hipótese: conjunto de dados do serviço de transfusão de sangue

Tabela A.18: Conjunto de dados do serviço de transfusão de sangue, teste de hipótese com significância de 10% para o algoritmo K-*Means* Linear.

Algoritmo K-*Means* Linear	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	+	=
		AKNN	+	+	=
			AKNC	+	−
				Stk1	−
					StkN

Fonte: Autoria própria

Tabela A.19: Conjunto de dados do serviço de transfusão de sangue, teste de hipótese com significância de 10% para o algoritmo RCL.

Algoritmo RCL	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	+	=
		AKNN	+	+	=
			AKNC	=	−
				Stk1	−
					StkN

Fonte: Autoria própria

Tabela A.20: Conjunto de dados do serviço de transfusão de sangue, teste de hipótese com significância de 10% para o algoritmo K-*Means* Híbrido.

Algoritmo K-*Means* Híbrido	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	=	=	+
		AKNN	=	=	+
			AKNC	=	+
				Stk1	+
					StkN

Fonte: Autoria própria

Tabela A.21: Conjunto de dados do serviço de transfusão de sangue, teste de hipótese com significância de 10% para o algoritmo MeSCH.

Algoritmo MeSCH	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	=	+
		AKNN	+	=	+
			AKNC	−	=
				Stk1	+
					StkN

Fonte: Autoria própria

A.7 Teste de hipótese: Conjunto de dados de compressão do concreto

Tabela A.22: Conjunto de dados de compressão do concreto, teste de hipótese com significância de 10% para o algoritmo K-*Means* Linear.

Algoritmo K-*Means* Linear	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	=	=
		AKNN	+	=	=
			AKNC	−	−
				Stk1	=
					StkN

Fonte: Autoria própria

Tabela A.23: Conjunto de dados de compressão do concreto, teste de hipótese com significância de 10% para o algoritmo RCL.

Algoritmo RCL	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	+	=
		AKNN	+	+	=
			AKNC	−	−
				Stk1	−
					StkN

Fonte: Autoria própria

Tabela A.24: Conjunto de dados de compressão do concreto, teste de hipótese com significância de 10% para o algoritmo K-*Means* Híbrido.

Algoritmo K-*Means* Híbrido	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	=	=
		AKNN	+	=	=
			AKNC	=	−
				Stk1	=
					StkN

Fonte: Autoria própria

Tabela A.25: Conjunto de dados de compressão do concreto, teste de hipótese com significância de 10% para o algoritmo MeSCH.

Algoritmo MeSCH	AAle	AKNN	AKNC	Stk1	StkN
	AAle	+	+	+	=
		AKNN	+	+	−
			AKNC	−	−
				Stk1	−
					StkN

Fonte: Autoria própria

A.8 Teste de hipótese: conjunto de dados de bicicletas

Tabela A.26: Conjunto de dados de bicicletas, teste de hipótese com significância de 10% para o algoritmo K-*Means* Linear.

Algoritmo K-*Means* Linear	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	−	=	=
		AKNN	=	=	=
			AKNC	=	=
				Stk1	=
					StkN

Fonte: Autoria própria

Tabela A.27: Conjunto de dados de bicicletas, teste de hipótese com significância de 10% para o algoritmo RCL.

Algoritmo RCL	AAle	AKNN	AKNC	Stk1	StkN
	AAle	+	+	=	−
		AKNN	+	−	−
			AKNC	−	−
				Stk1	−
					StkN

Fonte: Autoria própria

Tabela A.28: Conjunto de dados de bicicletas, teste de hipótese com significância de 10% para o algoritmo K-*Means* Híbrido.

Algoritmo K-*Means* Híbrido	AAle	AKNN	AKNC	Stk1	StkN
	AAle	+	=	=	=
		AKNN	=	−	=
			AKNC	−	−
				Stk1	=
					StkN

Fonte: Autoria própria

Tabela A.29: Conjunto de dados de bicicletas, teste de hipótese com significância de 10% para o algoritmo MeSCH.

Algoritmo MeSCH	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	+	+
		AKNN	+	+	+
			AKNC	=	−
				Stk1	−
					StkN

Fonte: Autoria própria

A.9 Teste de hipótese: conjunto de dados de abalone

Tabela A.30: Conjunto de dados de abalone, teste de hipótese com significância de 10% para o algoritmo K-*Means* Linear.

Algoritmo K-*Means* Linear	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	+	=
		AKNN	+	=	=
			AKNC	−	−
				Stk1	−
					StkN

Fonte: Autoria própria

Tabela A.31: Conjunto de dados de abalone, teste de hipótese com significância de 10% para o algoritmo RCL.

Algoritmo RCL	AAle	AKNN	AKNC	Stk1	StkN
	AAle	+	+	=	=
		AKNN	=	−	−
			AKNC	−	−
				Stk1	=
					StkN

Fonte: Autoria própria

Tabela A.32: Conjunto de dados de abalone, teste de hipótese com significância de 10% para o algoritmo K-*Means* Híbrido.

Algoritmo K-*Means* Híbrido	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	=	=
		AKNN	+	=	=
			AKNC	=	−
				Stk1	=
					StkN

Fonte: Autoria própria

Tabela A.33: Conjunto de dados de abalone, teste de hipótese com significância de 10% para o algoritmo MeSCH.

Algoritmo MeSCH	AAle	AKNN	AKNC	Stk1	StkN
	AAle	=	+	−	=
		AKNN	+	−	−
			AKNC	−	−
				Stk1	+
					StkN

Fonte: Autoria própria

Apêndice B

Tabelas dos testes de hipóteses dos algoritmos

B.1 Teste de hipótese: legenda para os símbolos

Tabela B.1: Legenda para os testes de hipótese

Símbolo	Descrição
$-$	Comparar que o método da linha é menor que o método da coluna teve um *p-valor* (nível descritivo ou probabilidade de significância) menor que o caso contrário ($+$) e o *p-valor* foi menor que a significância.
$+$	Comparar que o método da linha é maior que o método da coluna teve um *p-valor* menor que o caso contrário ($-$) e o *p-valor* foi menor que a significância.
$=$	O menor *p-valor* das comparações ($+$ ou $-$) foi maior ou igual que a significância.

Fonte: Autoria própria

B.2 Teste de hipótese: conjunto de dados de eficiência energética

Tabela B.2: Teste de hipótese: conjunto de dados de eficiência energética, teste de hipótese com significância de 10%.

K-*Means* Híbrido	MoSCH	K-*Means* Linear	RCL
K-*Means* Híbrido	=	+	=
	MoSCH	+	=
		K-*Means* Linear	−
			RCL

Fonte: Autoria própria

B.3 Teste de hipótese: conjunto de dados Auto MPG

Tabela B.3: Teste de hipótese: conjunto de dados Auto MPG, teste de hipótese com significância de 10%.

K-*Means* Híbrido	MoSCH	K-*Means* Linear	RCL
K-*Means* Híbrido	=	=	=
	MoSCH	=	+
		K-*Means* Linear	=
			RCL

Fonte: Autoria própria

B.4 Teste de hipótese: conjunto de dados de consumo de álcool

Tabela B.4: Teste de hipótese: conjunto de dados de consumo de álcool, teste de hipótese com significância de 10%.

	K-*Means* Híbrido	MoSCH	K-*Means* Linear	RCL
K-*Means* Híbrido		+	−	=
MoSCH			−	−
K-*Means* Linear				+
RCL				

Fonte: Autoria própria

B.5 Teste de hipótese: conjunto de dados de preços de casas

Tabela B.5: Teste de hipótese: conjunto de dados de preços de casas, teste de hipótese com significância de 10%.

	K-*Means* Híbrido	MoSCH	K-*Means* Linear	RCL
K-*Means* Híbrido		−	+	=
MoSCH			+	=
K-*Means* Linear				−
RCL				

Fonte: Autoria própria

B.6 Teste de hipótese: conjunto de dados do serviço de transfusão de sangue

Tabela B.6: Teste de hipótese: conjunto de dados do serviço de transfusão de sangue, teste de hipótese com significância de 10%.

K-*Means* Híbrido	MoSCH	K-*Means* Linear	RCL
K-*Means* Híbrido	=	=	−
	MoSCH	=	−
		K-*Means* Linear	−
			RCL

Fonte: Autoria própria

B.7 Teste de hipótese: conjunto de dados de compressão do concreto

Tabela B.7: Teste de hipótese: conjunto de dados de compressão do concreto, teste de hipótese com significância de 10%.

K-*Means* Híbrido	MoSCH	K-*Means* Linear	RCL
K-*Means* Híbrido	−	−	+
	MoSCH	=	+
		K-*Means* Linear	+
			RCL

Fonte: Autoria própria

B.8 Teste de hipótese: conjunto de dados do serviço de bicicletas

Tabela B.8: Teste de hipótese: conjunto de dados de bicicletas, teste de hipótese com significância de 10%.

	K-*Means* Híbrido	MoSCH	K-*Means* Linear	RCL
K-*Means* Híbrido		−	−	=
MoSCH			+	=
K-*Means* Linear				=
RCL				

Fonte: Autoria própria

B.9 Teste de hipótese: conjunto de dados de abalone

Tabela B.9: Teste de hipótese: conjunto de dados de abalone, teste de hipótese com significância de 10%.

	K-*Means* Híbrido	MoSCH	K-*Means* Linear	RCL
K-*Means* Híbrido		=	=	=
MoSCH			−	=
K-*Means* Linear				=
RCL				

Fonte: Autoria própria