UNIVERSIDADE FEDERAL DE SÃO CARLOS
DEPARTAMENTO DE COMPUTAÇÃO
ENGENHARIA DE COMPUTAÇÃO

Júlia Almeida Modesto

Agrupamento em conjunto de dados com múltiplos atributos via redução de dimensionalidade e detecção de comunidades

São Carlos - SP

2024

Júlia Almeida Modesto

Agrupamento em conjunto de dados com múltiplos atributos via redução de dimensionalidade e detecção de comunidades

Trabalho de Conclusão de Curso apresentado ao curso de Engenharia de Computação da Universidade Federal de São Carlos, como requisito parcial para a obtenção do título de Bacharel em Engenharia de Computação.

Orientação Prof. Dr. Alan Demétrius Baria Valejo

São Carlos - SP

2024

Dedico este trabalho a todos que me apoiaram em momentos desafiadores da graduação. Não foi fácil, mas eu consegui!

Agradecimentos

Primeiramente, agradeço à minha avó Cleusa por ter me dado a oportunidade de estudar nas melhores escolas e, assim, ter capacidade de entrar em uma das melhores faculdades do país. Agradeço também pelo apoio durante esses 5 anos de graduação, tenho certeza que sem a ajuda dela não seria a pessoa que sou e muito menos teria conquistado tudo o que tenho hoje. Obrigada, vovó!

Agradeço à minha mãe e à minha avó Nice por serem a minha fonte de inspiração e persistência, por não terem me deixado desistir quando a dificuldade bateu na porta e por terem sido meu ombro amigo quando precisei. Tenho certeza que todos os ensinamentos delas quanto a sobreviver sozinha foram cruciais para que minha jornada ocorresse de forma mais tranquila, mesmo com alguns perrengues no meio do caminho.

Agradeço a todos os meus amigos que foram fonte de carinho e felicidade. Ao Bruno, obrigada por me ensinar tanto e por ser meu parceiro das madrugadas acordadas, por estar comigo em momentos tão difíceis e por ser meu companheiro em tantas ocasiões. À Sara, por ser minha amiga confidente, por me fazer rir quando eu mais precisava e por fazer eu me sentir mais leve no dia a dia. Ao Italo por me acompanhar em tantas aventuras no mercado de trabalho, por ser meu companheiro da cafeteria da esquina e por me ajudar tanto em diversas matérias durante a graduação. Ao meu namorado, Guilherme, que mesmo longe sempre fez eu me sentir em casa e enxugou minhas lágrimas por diversas vezes. Às demais pessoas que não citei diretamente, porém que foram muito importantes durante toda a jornada, meus mais sinceros agradecimentos.

Aos meus professores do colégio, Nixon, Marcela, Credidio, Ana Márcia, Alexandre, Catarina, Marcão, Trota, Milton, Elisana, Miriam, Dutra, Ambrósio, Ivetinha e Miro, que são as pessoas mais incríveis que eu tive o prazer de conhecer e de aprender tanto. Que saudades que eu senti de vocês durante esses 5 anos de faculdade. Muito obrigada por todos os ensinamentos, é por causa de vocês que serei engenheira da computação em uma universidade federal. Nunca serei capaz de expressar completamente minha gratidão. Aos professores da UFSCar, gostaria de agradecer àqueles que foram luz no fim do túnel e que pegaram na minha mão para me levar mais longe, não só na vida acadêmica quanto na vida pessoal.

Também agradeço imensamente ao professor Dr. Alan Demétrius Baria Valejo, que topou ser meu orientador e fez um trabalho excepcional durante o desenvolvimento deste trabalho de conclusão de curso. Conseguimos cumprir todas as metas quinzenais e sinto que se não fosse pela organização do cronograma, feito por ele, eu estaria completamente perdida. Muito obrigada, professor!

"Por vezes sentimos que aquilo que fazemos não é senão uma gota de água no mar. Mas o mar seria menor se lhe faltasse uma gota"

(Madre Teresa de Calcuta)

Resumo

Este trabalho insere-se no campo do Aprendizado de Máquina e centra-se no desafio do Agrupamento em conjuntos de dados com múltiplas variáveis. Abordando a questão da Maldição da Dimensionalidade, uma dificuldade frequente em dados de alta dimensão, o estudo enfoca em técnicas de Redução de Dimensionalidade e Detecção de Comunidade. A Maldição da Dimensionalidade refere-se à deterioração do desempenho de certos algoritmos de aprendizado conforme a dimensionalidade dos dados aumenta. Poucos estudos abordam as implicações do agrupamento baseado em grafos nesse cenário, destacando a necessidade de investigar como técnicas de agrupamento podem ser adaptadas para lidar efetivamente com conjuntos de dados de alta dimensão. Através da aplicação de Algoritmos de Detecção de Comunidades em conjunto com a Construção de Grafos, o estudo visa explorar métodos eficientes para lidar com grandes volumes de dados. Os experimentos realizados mostraram que, em conjuntos de dados com um grande número de amostras, a utilização de técnicas de Redução de Dimensionalidade, em particular, resultou em um desempenho superior na identificação de agrupamentos significativos. Este resultado enfatiza a importância da Redução de Dimensionalidade como uma ferramenta crucial para superar os desafios impostos pela Maldição da Dimensionalidade em ambientes de Aprendizado de Máquina. Este trabalho pode auxiliar futuras investigações e aplicações práticas em análises de dados complexos e multidimensionais.

Palavras-chave: Aprendizado da Máquina; Algoritmos de Agrupamento; Construção de Grafos; Detecção de Comunidade; Maldição da Dimensionalidade; Redução de Dimensionalidade;

Abstract

This work falls within the field of Machine Learning and focuses on the challenge of Clustering in datasets with multiple variables. Addressing the issue of the Curse of Dimensionality, a common difficulty in high-dimensional data, the study focuses on Dimensionality Reduction and Community Detection techniques. The Curse of Dimensionality refers to the deterioration of the performance of certain learning algorithms as the dimensionality of the data increases. Few studies address the implications of graph-based clustering in this scenario, highlighting the need to investigate how clustering techniques can be adapted to effectively deal with high-dimensional datasets. Through the application of Community Detection Algorithms in conjunction with Graph Construction, the study aims to explore efficient methods for handling large volumes of data. The experiments conducted showed that, in datasets with a large number of samples, the use of Dimensionality Reduction techniques, in particular, resulted in superior performance in identifying significant clusters. This result emphasizes the importance of Dimensionality Reduction as a crucial tool for overcoming the challenges posed by the Curse of Dimensionality in Machine Learning environments. This work can assist future investigations and practical applications in complex and multidimensional data analysis.

Keywords: Machine Learning; Traditional Clustering Algorithms; Graph Construction; Community Detection; Curse of Dimensionality.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

No cenário atual, com o crescente número de usuários com acesso a internet, o uso de variados dispositivos eletrônicos - como IoTs, celulares, computadores e assistentes virtuais - e a crescente capacidade de armazenamento têm resultado em conjuntos de dados cada vez mais complexos e multidimensionais. Embora representem uma fonte importante de informação, analisar e treinar modelos utilizando dados tão complexos traz um obstáculo: a Maldição da Dimensionalidade, a qual refere-se a uma série de problemas que surgem ao trabalhar com dados de alta dimensionalidade, tais como a diminuição da distância entre os pontos, aglomeração dos dados e dificuldade de visualização. Para lidar com essas dificuldades, o Aprendizado de Máquina (AM) possui algumas técnicas que auxiliam na diminuição da dimensionalidade para que, assim, seja possível treinar modelos com dados complexos.

O AM, subárea da Inteligência Artificial (IA), tem como objetivo principal desenvolver algoritmos que possam reconhecer padrões e tomar decisões a partir de dados, sem que seja necessária uma programação específica e explícita (MITCHELL, 1997). A aprendizagem de um modelo começa após a definição de um problema. Com isso, coleta-se os dados necessários (previamente tratados) para o treinamento agrupados na forma de uma tabela. A partir dessa etapa, o cientista de dados escolhe o tipo de modelo que será utilizado, fornece os dados coletados e, então, permite que o modelo treine a si mesmo com a intenção de encontrar padrões e criar previsões em cima dos resultados obtidos (BROWN, 2021).

No AM destacam-se dois tipos principais de aprendizado: o supervisionado e o não-supervisionado. No aprendizado supervisionado, os modelos são treinados com dados já etiquetados (ou rotulados), o que facilita a capacidade do modelo em fazer previsões baseadas em respostas conhecidas. Por outro lado, no aprendizado não-supervisionado, o modelo deve identificar padrões e estruturas inerentes nos dados sem orientações prévias.

Dentro do aprendizado não-supervisionado, um segmento importante é o agrupamento (ou clusterização). Este processo envolve a classificação de objetos com base em sua similaridade ou critérios de proximidade. Exemplos de algoritmos tradicionais de agrupamento incluem o K-means e o Agrupamento Aglomerativo (SINAGA, 2020).

Além destes algoritmos, os baseados em grafos ganham destaque por conseguir representar as relações entre os objetos de forma mais clara e objetiva. Os grafos são, resumidamente, estruturas matemáticas com dois elementos principais: os nós e as arestas. No caso do agrupamento, os nós representam os dados e as arestas as relações entre eles. Para construir um grafo, é necessário utilizar-se de algoritmos que considerem os

atributos e a estrutura dos dados, podendo estes basear-se de aspectos como similaridade ou dissimilaridade (euclidiana, Manhattan, entre outras); assim, será possível ter a estrutura apropriada para realizar a *detecção de comunidades*.

A detecção de comunidades é uma técnica que visa identificar grupos ou comunidades de nós em um grafo, onde os nós dentro de uma mesma comunidade são mais fortemente conectados entre si do que com os nós de outras comunidades. Essa técnica é fundamental em diversos campos, como análise de redes sociais, biologia computacional, ciência da informação, entre outros. Algoritmos de detecção de comunidades exploram padrões de conexões dentro do grafo para identificar esses agrupamentos, auxiliando na compreensão da estrutura e organização dos dados.

Para isso, algoritmos de detecção de comunidades, como o *Fastgreedy*, são ferramentas poderosas no contexto de agrupamento de dados. Eles permitem não apenas uma análise mais profunda e significativa das relações inerentes nos dados, mas também facilita a visualização e a compreensão de grandes grafos, tornando-se indispensável em muitas aplicações práticas e teóricas.

No contexto do agrupamento de dados, a *Maldição da Dimensionalidade* surge quando tem-se um conjunto de dados com um grande número de atributos, dificultando, assim, o processo de aprendizagem. Um aglomerado de dados com atributos irrelevantes prejudica a performance e acurácia dos algoritmos, e, consequentemente, causa sobreposição e dificuldade na detecção de comunidades. Além disso, aumentar a quantidade de atributos nos dados também aumenta o processamento necessário para realizar a identificação de padrões.

Com isso, utilizar de técnicas que diminuam a quantidade de atributos, como o uso da técnica *Principal Components Analysis* (PCA), tornou-se uma opção viável para que seja possível tratar dados e treinar modelos sem interferir na qualidade do resultado final.

1.1 Objetivos

1.1.1 Objetivo Geral

Este trabalho de conclusão de curso tem por objetivo apresentar um estudo sobre o agrupamento baseado em grafos em conjunto de dados com múltiplas variáveis via redução de dimensionalidade e detecção de comunidades. O agrupamento será realizado usando o algoritmo de detecção de comunidades fastgreedy, bem como técnicas como PCA para a reduzir a dimensionalidade.

1.1.2 Objetivos específicos

Os objetivos específicos podem ser elencados em:

- Contribuir com a literatura sobre o desempenho dos algoritmos de construção de grafos em conjuntos de alta dimensionalidade;
- Avaliar um algoritmo de detecção de comunidades em diferentes grafos gerados a partir de dados de alta dimensão com e sem redução de dimensionalidade;
- Analisar os resultados obtidos, comparando os algoritmos de construção de grafos com detecção de comunidades com algoritmos tradicionais (*baseline*) dadas diferentes dimensões, aplicando a redução de dimensionalidade para ambos os casos.

1.2 Organização do trabalho

Este trabalho será dividido em cinco capítulos. No Capítulo 1 são apresentados a proposta de pesquisa e os objetivos que serão explorados. O Capítulo 2 abrange a fundamentação teórica necessária para cumprir com o objetivo proposto, passando pelos conceitos de AM, agrupamento, redução de dimensionalidade, além de justificativas acerca dos algoritmos utilizados e das métricas escolhidas para avaliar os resultados. No Capítulo 3 exibe-se a metodologia utilizada para a execução dos experimentos. O Capítulo 4 destina-se à análise dos resultados obtidos e comparação entre os algoritmos com diferentes dimensionalidades. Por fim, o Capítulo 5 expõe as conclusões do trabalho.

2 Fundamentação Teórica

Neste capítulo, serão apresentados os principais conceitos utilizados para embasamento do trabalho e experimentos realizados.

2.1 Aprendizado de Máquina

Aprendizado de Máquina (AM) é um subcampo da Inteligência Artificial (IA) definida como a capacidade de uma máquina “aprender” e imitar o comportamento de seres humanos inteligentes. O objetivo principal de uma IA é criar modelos computacionais que demonstrem "comportamentos inteligentes"semelhantes ao de humanos quando resolvem problemas complexos. O cientista de dados escolhe o tipo de modelo que será utilizado, fornece os dados coletados e, então, permite que o modelo treine a si mesmo com a intenção de encontrar padrões e criar previsões em cima dos resultados obtidos(BROWN, 2021).

Uma abordagem popular em aprendizagem de máquina é o aprendizado indutivo, processo através do qual um modelo aprende a generalizar a partir de exemplos específicos, formando regras ou padrões que podem ser aplicados a novos dados. O aprendizado indutivo pode ser dividido em duas categorias principais: Aprendizado Supervisionado e Não Supervisionado, conforme ilustrado na Figura 1.

Figura 1 – Estrutura do aprendizado indutivo.

Fonte: (MONTEIRO, 2018).

2.1.1 Aprendizagem Supervisionada

Os algoritmos de aprendizado de máquina tratam cada instância de um conjunto de dados como uma coleção de características, podendo estas serem binárias, categóricas ou contínuas. Se estas instâncias estiverem rotuladas, esse tipo de aprendizado é chamado de **aprendizado supervisionado**.

O aprendizado supervisionado envolve treinar o modelo com dados rotulados e testá-lo com dados não rotulados. Após a coleta de dados inicial, o conjunto de dados é dividido em dados de teste e treinamento e, então, os dados são pré-processados. O modelo é treinado para aprender os padrões associados a cada rótulo e, por fim, espera-se que este retorne os rótulos esperados, realizando previsões nos dados de teste (DRIDI, 2021).

2.1.2 Aprendizagem Não-Supervisionada

Diferentemente do aprendizado supervisionado, o algoritmo de aprendizado não supervisionado encontra padrões, podendo ser grupo, em um conjunto de dados não rotulados com base na estrutura encontrada no próprio conjunto. Quando estamos realizando uma tarefa de agrupamento, os grupos representam dados não rotulados que demonstram semelhanças entre si, mesmo que não possuam um rótulo explicitamente definido (JONES, 2017). No presente documento, utilizaremos esta abordagem como foco principal, utilizando-se principalmente de técnicas de agrupamento e detecção de comunidades, as quais serão especificadas nos próximos tópicos.

2.2 Algoritmos de Agrupamento Tradicionais

Algoritmos de agrupamento fazem parte da categoria de aprendizado não-supervisionado e são utilizados para agrupar um conjunto de dados em grupos semelhantes - chamados de grupos (ou*grupos*). O objetivo é criar grupos em que os membros tenham alta semelhança interna e baixa semelhança com outros membros de outros grupos.

Os algoritmos de agrupamento tradicionais podem ser divididos em duas categorias: particional e hierárquico. O primeiro divide o conjunto de dados em um número específico de grupos sem estabelecer qualquer hierarquia entre eles e geralmente o número de grupos precisa ser estabelecido antecipadamente; já o segundo organiza os dados em uma estrutura de árvore, revelando relações hierárquicas entre os pontos de dados. Abaixo, serão descritos, de maneira resumida, quais métodos cada algoritmo utiliza a fim de agrupar os dados, conforme citado anteriormente.

2.2.1 Algoritmo *K-means*

O algoritmo *k-means* é o método de agrupamento particional mais conhecido e utilizado devido a sua simplicidade. Este algoritmo necessita que a quantidade de grupos seja definida à priori pelo usuário, gerando k grupos e reunindo os dados de acordo com os padrões encontrados.(JAIN, 2010)

Porém, o algoritmo apresenta algumas limitações, como sensibilidade a *outliers*, necessidade de normalização das variáveis, em caso de escalas desiguais, dificuldade de aprender com dados de alta dimensionalidade, e até mesmo definir a quantidade de grupos pelo usuário também pode ser considerada uma limitação (SINAGA, 2020).

Portanto, os passos necessários para aplicar o algoritmo *k-means* são:

1. **Escolha o número de grupos**: Determine o número k de grupos que serão identificados no conjunto de dados.
2. **Inicialize os centroides**: Selecione aleatoriamente k pontos dos dados como os centroides iniciais dos clusters.
3. **Atribuição dos pontos aos grupos**: Para cada ponto no conjunto de dados, calcule a distância desse ponto a cada um dos k centroides.
4. **Recálculo dos Centroides**: Uma vez que todos os pontos foram atribuidos a um grupo, recalcule a posição de cada centroide
5. **Repita**: Repita os passos 3 e 4 até que os centroides não mudem significativamente entre as iterações.

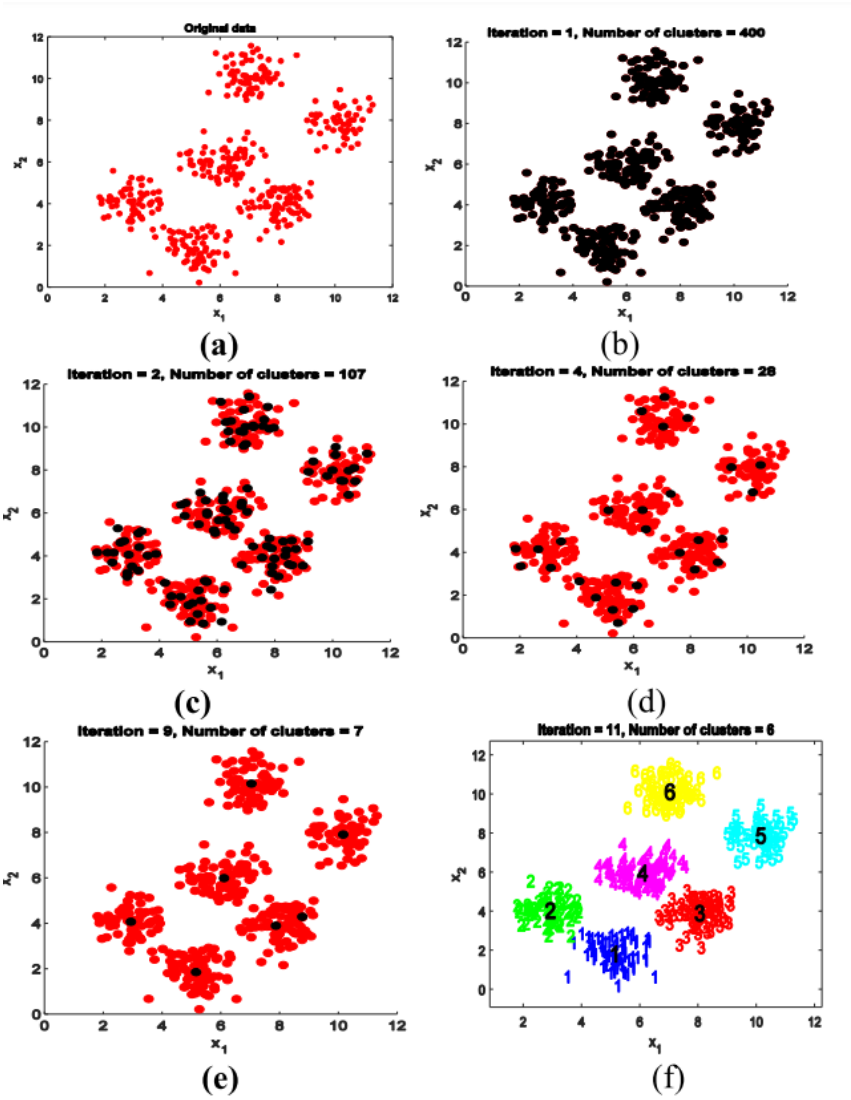
A Figura 2 mostra a aplicação do algoritmo *K-means* sendo: (a) conjunto de dados original, (b) 400 grupos, (c) 107 grupos (d) 28 grupos (e) 27 grupos e (f) 6 grupos, sendo o final do agrupamento obtido pelo algoritmo *K-means* na convergência (SINAGA, 2020).

2.2.2 Agrupamento Hierárquico

O agrupamento hierárquico (ou aglomerativo) é um dos principais métodos de agrupamento usados em análise de dados. Diferente do K-Means, que é um método de agrupamento particional e não hierárquico, o agrupamento aglomerativo constrói uma hierarquia de grupos de forma “de baixo para cima” (bottom-up).

O escolhido para ser aplicado neste trabalho foi o *Ward* (**??**), que utiliza o método de agrupamento distintivo; matematicamente falando, o algoritmo é baseado no critério de soma dos quadrados, com o objetivo de formar grupos que diminuem a dispersão à medida que os grupos são combinados, além de explorar o espaço euclidiano em dados de alta dimensionalidade em busca de grupos.

Figura 2 – Ilustração do algoritmo *K-means*.



Fonte: (SINAGA, 2020)

2.2.3 Algoritmos de Agrupamento Baseados em Grafos

2.2.3.1 Definições iniciais e a importância dos grafos

Os algoritmos de agrupamento para detecção de comunidades tem por objetivo encontrar estruturas de comunidades interconectadas por grafos. Dados representados por este tipo de estrutura compõem-se de um conjunto de nós e um conjunto de arestas que conectam esses nós. Os nós representam os objetos, enquanto as arestas representam as relações existentes entre estes. Assim, é possível definir, de forma matemática, que um grafo $G(V, A)$ representa um conjunto finito $V = v_1, v_2, \ldots v_n$, no qual os $n = |V|$ elementos são denominados **nós**, juntamente com um conjunto $A = A_1, A_2, ..., A_m$ de arestas, tal que uma aresta que liga os nós v_i e $v_j \in V$ pode ser denotada por $a_{i,j}$. Com isso, dois nós são ditos adjacentes ou vizinhos se estão conectados por uma aresta. Para realizar a detecção de comunidade, os algoritmos utilizados para este fim precisam que os

dados estejam organizados em forma de grafos.

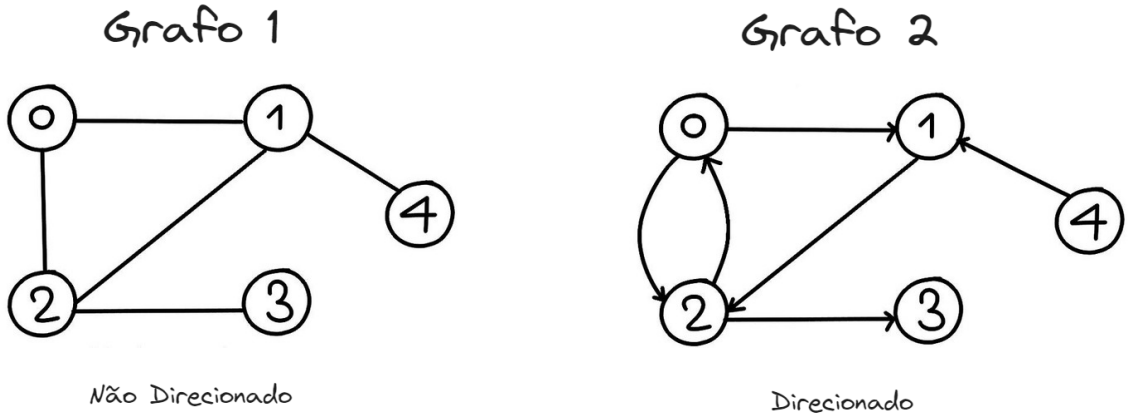
Existem alguns casos em que os dados já são extraídos em forma de grafos, porém, em alguns contextos, os dados não são naturalmente relacionais e, para essas situações, é necessário escolher algoritmos de construção de grafos específicos para transformar a estrutura e relação dos dados (ZHU, 2005).

Para realizar a transformação, existem inúmeros tipos de grafos, os mais comuns são:

- **Direcionados**: as arestas possuem direção e orientação específica, cada aresta começa na sua ponta inicial e termina na sua ponta final;
- **Não direcionados**: a conexão entre os nós é bidirecional, ou seja, cada uma de suas arestas é antiparalela a alguma outra aresta: para cada aresta a_vw, o grafo também tem a aresta a_wv.;
- **Ponderados**: Cada aresta possui um peso (força de associação), e influencia no cálculo ou construção do grafo em si;
- **Não ponderados**: todas as arestas possuem o mesmo peso;
- Entre outros tipos de grafos com atributos específicos (FEOFILOFF, 2017).

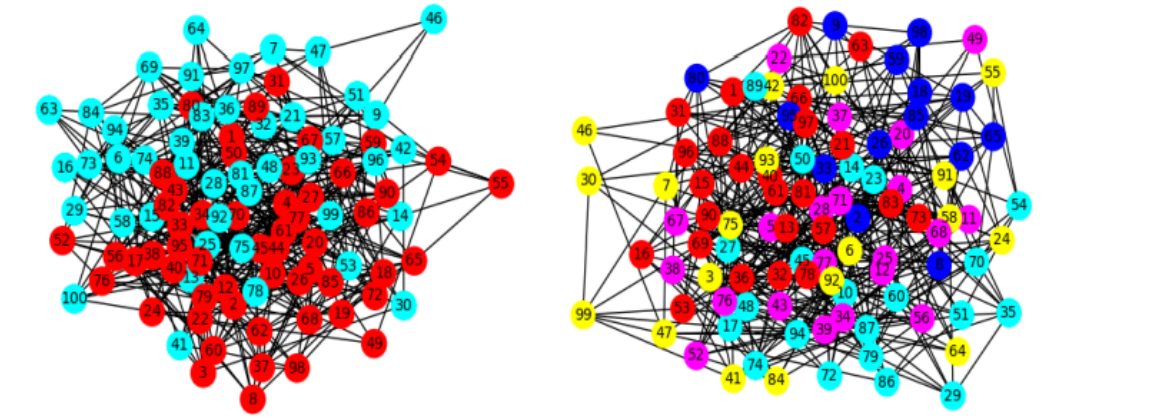
Na Figura 3 o grafo 1 representa um grafo não direcionado e o 2, um grafo direcionado. Já na Figura 4 é possível observar o agrupamento utilizando grafos para grafos sociais; o exemplo (a) foi possível agrupar as pessoas em duas comunidades, no exemplo (b), agrupou-se 100 usuários em cinco comunidades distintas.

Figura 3 – Exemplo de grafos direcionado e não direcionado.



Fonte: (DINH, 2020).

Figura 4 – Exemplo de grafos para agrupamento de grafo social.



(a) Grafo com Duas Comunidades (b) Grafo com Cinco Comunidades

Fonte: (MAJEED; RAUF, 2020).

2.2.3.2 *k-Nearest Neighbours (k-NN)*

Embora seja um algoritmo originalmente desenvolvido para aprendizado supervisionado, o *k-Nearest Neighbours (k-NN)* pode ser utilizado como medida de proximidade em um grafo de comunidades (GUO et al., 2003).

Para realizar a análise, dado um conjunto de amostras $X = \{x_1, \ldots, x_n\}$ e $i \neq j$, o algoritmo considera que o vizinho mais próximo de x_i é x_j se este detém a menor distância possível a x_i. Um cálculo de distância muito utilizado é a *distância Euclidiana*.

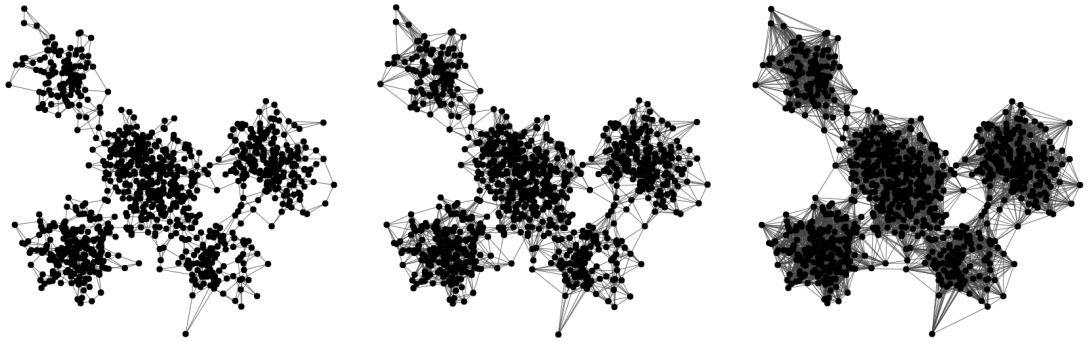
Com isso, o algoritmo k-NN utiliza as distâncias encontradas entre os objetos como base para construir um grafo, em que os k vizinhos mais próximos de cada x_i são conectados por arestas, representando as relações de proximidade.

Na Figura 5 gerou-se grafos utilizando o algoritmo k-NN para diferentes valores de k. Com isso, é possível visualizar que, no exemplo (a), um valor baixo para k pode gerar um grafo desconexo. Já nos exemplos (b) e (c), nota-se que, ao aumentar o valor de k, as arestas consequentemente tornam-se mais densas. No exemplo (c), entretanto, também observa-se que um nó é forçado a se conectar com os n vizinhos mais próximos com um valor de k muito alto, ocasionando, portanto, muitas ligações de nós com pouca similaridade.

2.2.3.3 *Mutual k-NN*

O algoritmo *Mutual k-NN* (Mk-NN) é derivado do *k-NN* citado acima; sua principal diferença se dá pela necessidade da regra de vizinhança cumprir-se mutualmente, ou seja, os k-vizinhos mais próximos mutualmente possuem alta probabilidade de pertencerem

Figura 5 – Exemplo de grafos gerados pelo k-NN dado um conjunto de dados com 1000 elementos

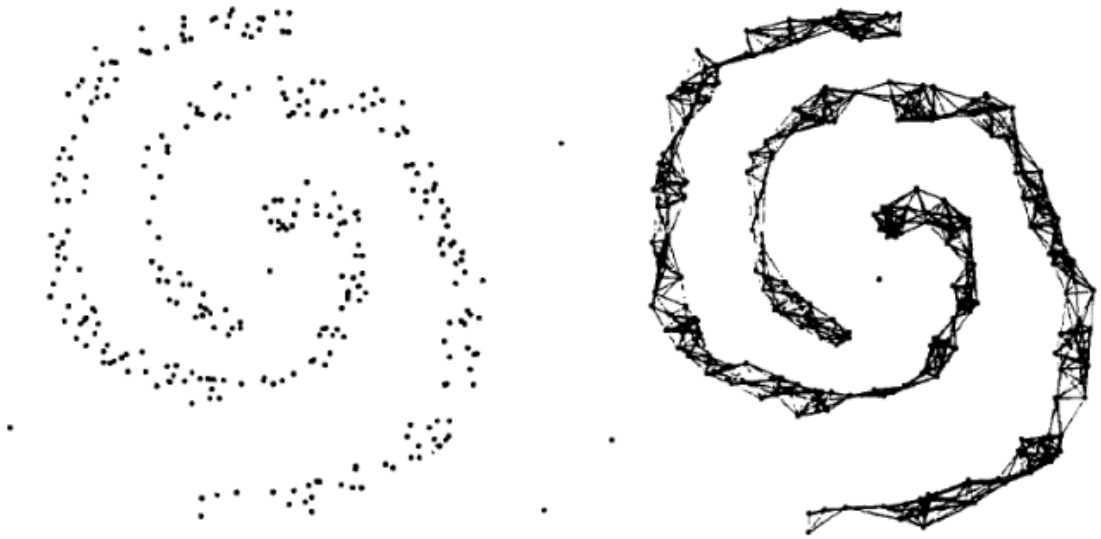


(a) k = 3 **(b)** k = 7 **(c)** k = 20

Fonte: desenvolvido pela autora

à mesma comunidade. Por isso, o Mutual k-NN é considerado mais limitado. Assim, o algoritmo k-NN é utilizado em contextos mais amplos por ser menos restritivo e conseguir identificar diversos grupos simultaneamente. Entretanto, o algoritmo Mutual k-NN é recomendado para identificar regiões de alta densidade de dados. A Figura 6 demonstra a criação de um grafo utilizando o Mk-NN para $k = 11$ em um determinado conjunto de dados.

Figura 6 – Grafo construído utilizando o Mk-NN com k = 11.



Fonte: (BRITO et al., 1997).

2.2.4 Algoritmos de Detecção de Comunidade

Em grafos complexos, comunidades representam a organização de nós em grupos reconhecidos pela existência de várias conexões entre nós dentro deste mesmo grupo, o que indica que provavelmente compartilham características comuns ou desempenham funções semelhantes no grafo. Porém, há poucas conexões entre nós de diferentes grupos. O processo de identificar tais grupos, portanto, é conhecido como Detecção de Comunidades, em que há identificação de estruturas naturais de comunidades em um grafo (FORTUNATO, 2010).

Um exemplo prático da aplicação da Detecção de Comunidades é observado em um grafo de relacionamento de alunos, em que agrupá-los por semelhança torna possível prever o comportamento dos membros do grafo (JUSTEL et al., 2014).

Os métodos de Detecção de Comunidades se diferenciam dos métodos de agrupamento tradicionais principalmente porque estes consideram apenas a similaridade de dados, utilizando medidas de distância, por exemplo. Já o primeiro considera, além da similaridade, a topologia, estrutura ou dinâmica do grafo. Com isso, é possível captar diversos padrões por meio da análise de funcionalidades e processos.

2.2.4.1 *Fastgreedy*

O algoritmo *Fastgreedy* é um método de Detecção de Comunidades hierárquico utilizado para particionar um grafo em comunidades. Este algoritmo pertence à família de aglomerativos, ou seja, ele começa com cada vértice em seu próprio grupo e, em seguida, gradualmente funde grupos adjacentes em etapas sucessivas.

O objetivo deste algoritmo é encontrar um agrupamento hierárquico que maximize uma medida de qualidade, muitas vezes baseada nas arestas ou pesos das conexões entre os vértices.

O algoritmo Fastgreedy é "greedy"(guloso) no sentido de que ele faz ótimas escolhas locais em cada etapa, visando maximizar a qualidade imediata da fusão. Ele é um algoritmo eficiente computacionalmente e funciona bem em muitos cenários, especialmente quando o objetivo é obter uma visão geral da estrutura hierárquica dos dados. É um dos vários algoritmos de agrupamento hierárquico disponíveis e é frequentemente utilizado em conjunto com outras técnicas para análise e visualização de grafos e grafos complexos (CLAUSET; NEWMAN; MOORE, 2004).

2.3 Medidas de avaliação

No aprendizado não-supervisionado, avaliar os resultados é um processo desafiador, visto que não há classes ou medidas objetivas claras de desempenho para orientar a

avaliação. Isso implica em confiar mais na interpretação subjetiva e na habilidade humana para validar os resultados obtidos pelos algoritmos de aprendizado não supervisionado. Como demonstrado por Hartigan (HARTIGAN, 1985), tem-se que diferentes algoritmos são corretos para diferentes propósitos, então não se pode afirmar que um é o melhor para tudo. Isto elucida a dificuldade em termos uma avaliação coerente dos resultados.

Assim, para validar um agrupamento é possível utilizar três tipos de critérios: relativos, internos e externos (JAIN; DUBES, 1988). Os índices baseados em critérios relativos comparam diversos agrupamentos para decidir qual deles é o melhor dado algum aspecto, baseando somente nos dados originais. Os que utilizam critérios internos são calculados utilizando-se exclusivamente dos atributos intrínsecos dos dados. Já o critério externo implica na comparação dos resultados com uma estrutura de referência previamente conhecida. Neste trabalho será utilizado o índice externo denominado Informação Mútua Normalizada (*Normalized Mutual Information*), uma vez que os experimentos serão realizados em dados sintéticos, ou seja, há uma partição de referência.

2.3.1 *Normalized Mutual Information*

A Informação Mútua Normalizada, *Normalized Mutual Information*, (NMI) é uma medida utilizada para avaliar a similaridade entre as partições geradas por um algoritmo de agrupamento em relação às classes reais dos dados, tendo sua base no conceito de informação mútua, que é uma medida estatística da dependência entre duas variáveis. Seu objetivo é fornecer um resultado entre 0 a 1, no qual 0 indica que não há similaridade mútua entre os conjuntos e 1 indica uma correlação perfeita (STREHL; GHOSH, 2002). O NMI é particularmente útil quando sabe-se a divisão dos dados previamente ou quando se avaliam dados artificiais que apresentam uma partição de referência conhecida.

O cálculo do NMI é realizado dessa forma: Levando em consideração dois conjuntos, predito (U) e previsto (V). Calcula-se a entropia (H) para ambos:

$$H(U) = -\sum_{i=1}^{|U|} P(i)log(P(i)) \tag{2.1}$$

$$H(V) = -\sum_{j=1}^{|V|} P(j)log(P(j)) \tag{2.2}$$

em que $P(i)$ é a probabilidade de um objeto aleatório de U ser atribuído à classe U_i, da mesma forma para $P(j)$. Após a realização do cálculo das entropias, calcula-se também o índice de informação mútua (MI) para os conjuntos predito e previsto:

$$MI(U,V) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} P(i,j)log(\frac{P(i,j)}{P(i)P'(j)}) \tag{2.3}$$

$P(i,j) = |U \cap V|/N$ sendo o número total de objetos. O MI pode ser reescrito da seguinte forma:

$$MI(U,V) = \sum_{i=1} |U| \sum_{j=1} |V| \frac{|Ui \cap Vj|}{N} log(\frac{N|Ui \cap Vj|}{|Ui||Vj|}) \tag{2.4}$$

O NMI pode ser escrito, portanto, como:

$$NMI(U,V) = \frac{MI(U,V)}{media(H(U), H(V))} \tag{2.5}$$

2.3.2 *Good Edges*

Good Edges foi o nome estabelecido para a medida dada pela proporção de arestas que ligam nós de categorias diferentes, referida como proporção de ϕ-arestas, apresentado por Ozaki (OZAKI et al., 2011). Assim, é possível com tal métrica analisar inicialmente o comportamento dos algoritmos de construção de grafos para os conjuntos de dados com alta dimensionalidade. Visando facilitar o entendimento, neste trabalho, foi feita a inversão da métrica original que mostra a quantidade de arestas erradas.

2.4 Dimensionalidade

Um dado pode ser descrito e armazenado como um conjunto de atributos. Em uma ficha médica, por exemplo, os dados de um paciente podem ser nome, idade, peso, doenças crônicas, tipo sanguíneo, dentre outras. Todas essas informações são atributos de um paciente. Outra nomenclatura utilizada para denominar atributos de um dado seria "dimensão", uma vez que um objeto com n atributos pode ser representado como um ponto com n dimensões (VLACHOS, 2010).

2.4.1 Alta Dimensionalidade

Na área de AM, a alta dimensionalidade diz respeito a conjuntos de dados que possuem um número de atributos muito maior do que o número de amostras do conjunto de dados (SIRIMONGKOLKASEM; DRIKVANDI, 2019). Conforme a dimensionalidade de dados aumenta, a tendência é que estes se tornem menos densos e, consequentemente, mais dispersos.

Lidar com dados de muitas dimensões é um problema que diversas áreas enfrentam periodicamente. Um exemplo disso está no domínio biomédico, em que conjuntos de dados possuem muito mais variáveis do que observações, o que leva a um crescimento exponencial nos bancos de dados. Além da biomedicina, muitas outras áreas da ciência enfrentam o

mesmo problema. Isso ocorre em processamento de imagens, análise de séries temporais, análise de texto automática, dentre muitos outros.

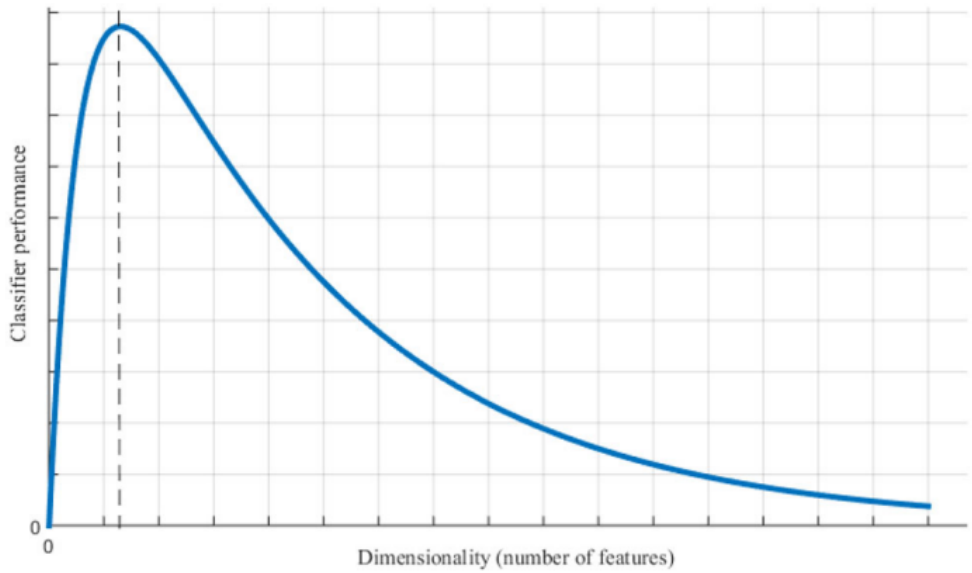
Com isso, reduzir o número de variáveis de entrada antes da aplicação de algum algoritmo de aprendizado de máquina é um método estatístico que vem se tornando cada vez mais frequente. A redução de dimensionalidade pode ser aplicada de duas maneiras:

- **Seleção de características:** mantém-se apenas as variáveis mais relevantes do conjunto de dados original.
- **Redução de dimensionalidade:** explora-se a redundância dos dados de entrada e encontra-se um conjunto menor de novas variáveis, sendo cada conjunto uma combinação das variáveis de entrada contendo essencialmente as mesmas informações que as variáveis de entrada.

2.4.2 Maldição da Dimensionalidade

A *Maldição da Dimensionalidade* descreve o problema em que o aumento de atributos leva a um aumento considerável da complexidade e, consequentemente, do custo computacional. Assim, ao contrário do que se imagina, o aumento do número de dimensões não implica diretamente em um aumento da performance do algoritmo. Costuma-se observar que, após um determinado "ponto ótimo", há uma queda na performance do modelo, conforme mostrado na Figura 7. Além disso, em conjuntos de dados com alta dimensionalidade, não é raro encontrar informações duplicadas ou irrelevantes para o classificador (JIA et al., 2022).

Figura 7 – Gráfico de performance vs número de atributos para um classificador.



Fonte: (JIA et al., 2022).

2.4.3 Redução da Dimensionalidade

Para lidar com os problemas apresentados, é possível aplicar uma redução na dimensionalidade dos sem perder informações e manter a boa performance do modelo. Como citado na seção 2.4.1, existem alguns métodos que podem ser utilizados, dentre eles, o algoritmo PCA.

2.4.3.1 Principal Analysis Component (PCA)

O *Principal Analysis Component* (PCA) é uma técnica de aprendizado não supervisionado utilizada para a redução de dimensionalidade. O objetivo dessa técnica consiste em reduzir a dimensionalidade dos dados mantendo a "variância máxima", comumente utilizada para grandes conjuntos de dados que contém muitas informações. É uma técnica estatística ortogonal que converte um conjunto de observações de variáveis relacionadas em um conjunto de valores não linearmente relacionados.

As vantagens do PCA incluem superar a duplicidade de características nos dados, obtenção de informações valiosas e explicação do alto contraste, fornecendo a melhor resolução, melhor visualização dos dados, redução de complexidade e aumento da eficiência computacional (HASAN; ABDULAZEEZ, 2021).

3 Metodologia

Este capítulo será dedicado a explicar os métodos utilizados para a criação dos conjuntos de dados sintéticos, algoritmos e ferramentas com o propósito de experimentação e obtenção de resultados. Os algoritmos foram elaborados na linguagem de programação Python[1], com o auxílio das bibliotecas do Scikit-learn[2] e do igraph[3].

3.1 Geração dos conjuntos de dados

Para os experimentos, utilizou-se de bases de dados (datasets) sintéticos feitas a partir do método *make blobs* da biblioteca *scikit-learn*. Essa ferramenta é frequentemente utilizada para criar conjuntos de dados para experimentos de agrupamento, teste de algoritmos de classificação e outras análises de dados. O *make blobs* gera dados distribuídos em grupos que são normalmente utilizados para simular conjuntos de dados com características de agrupamento. Cada grupo é formado em torno de um centro, e os dados são distribuídos em torno desses centros com uma variância que pode ser controlada pelo usuário. Os parâmetros utilizados na função foram:

- *samples*: Quantidade de amostras a serem geradas
- *features*: Quantidade de atributos para cada dado
- *clusters*: Quantidade de clusters gerados
- *std*: O desvio padrão dos grupos para cada cluster

Os demais parâmetros não foram especificados acima pois utilizaram os valores padrão da ferramenta e não serão relevantes para as análises experimentais.

A ferramenta começa partindo da geração de centros a partir do valor especificado pelo parâmetro *clusters*. Depois, em torno de cada centro, os pontos são gerados seguindo uma distribuição normal (ou Gaussiana) multivariada. O desvio padrão dessa distribuição pode ser ajustado usando o parâmentro *std*, permitindo coltrolar quão dispersos estão os pontos em torno de cada centro. Por fim, após a geração dos pontos, os dados serão embaralhados e isso garante que a ordem dos pontos não siga a ordem de geração dos grupos.

[1] <https://www.python.org/>
[2] <https://scikit-learn.org/stable/>
[3] <https://python.igraph.org/en/stable/>

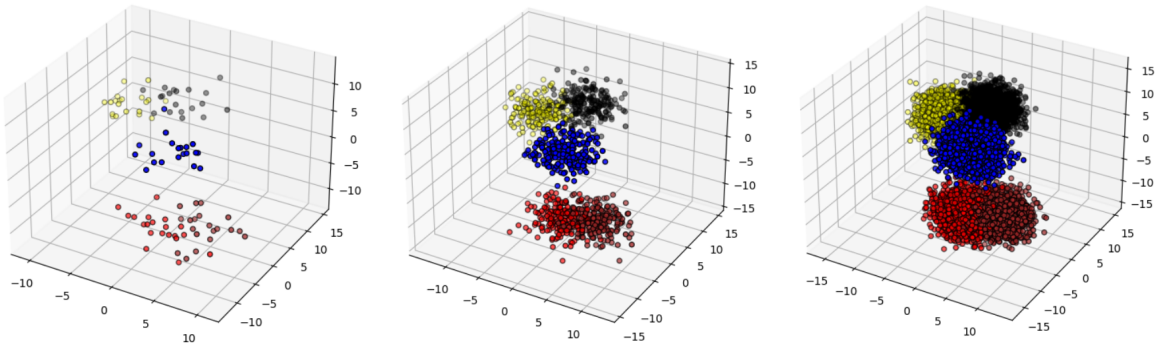
Para dados com alta dimensionalidade, um problema recorrente mencionado anteriormente se dá pela dificuldade em visualizar os dados. Alterando os valores dos parâmetros da função *create_dataset*, é possível observar o que cada um influencia na criação dos dados - mantendo apenas o parâmetro *features* igual a 3 em todas as variações.

Na seção a seguir, será demonstrado como cada alteração interfere na visualização dos dados.

3.1.1 Alterando a Quantidade de Amostras

A Figura 8 representa a variação na quantidade de amostras nos dados. Com isso, é possível observar que quando se varia apenas a quantidade de amostras, o volume de dados aumenta.

Figura 8 – Variando a quantidade de amostras do banco de dados.



(a) Amostras: 100 **(b)** Amostras: 1000 **(c)** Amostras: 10000

Fonte: desenvolvido pela autora

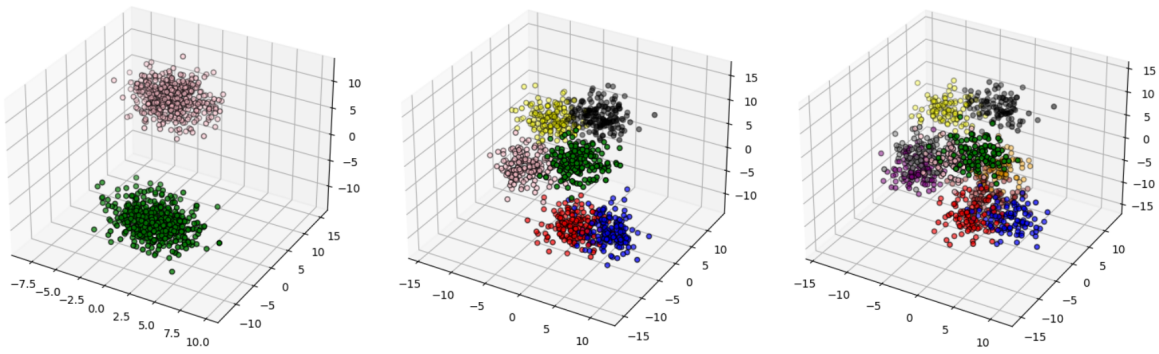
3.1.2 Alterando a Quantidade de *Clusters*

Com mais *clusters*, os dados tendem a ser distribuídos entre mais grupos, o que pode tornar mais difícil para algoritmos de agrupamento identificarem corretamente os grupos separados, especialmente se os *clusters* estiverem próximos ou se sobrepuserem. Além disso, com muitos *clusters*, a visualização pode se tornar confusa e menos intuitiva, o que não acontece quando tem-se um número menor de grupos. Na Figura 17 é possível observar que, com 2 a 6 *clusters* ainda é possível diferenciar visualmente grupos diferentes. Porém, ao aumentar para 10, ocorre a sobreposição de *clusters*, dificultando, portando, a visualização.

3.1.3 Alterando o Desvio Padrão

Variar o desvio padrão (std) dos *clusters* afeta a dispersão dos pontos de dados dentro de cada *cluster*, uma vez que o desvio padrão controla o quanto os pontos se

Figura 9 – Variando a quantidade de clusters do banco de dados.



(a) Clusters: 2 **(b)** Clusters: 6 **(c)** Clusters: 10

Fonte: desenvolvido pela autora

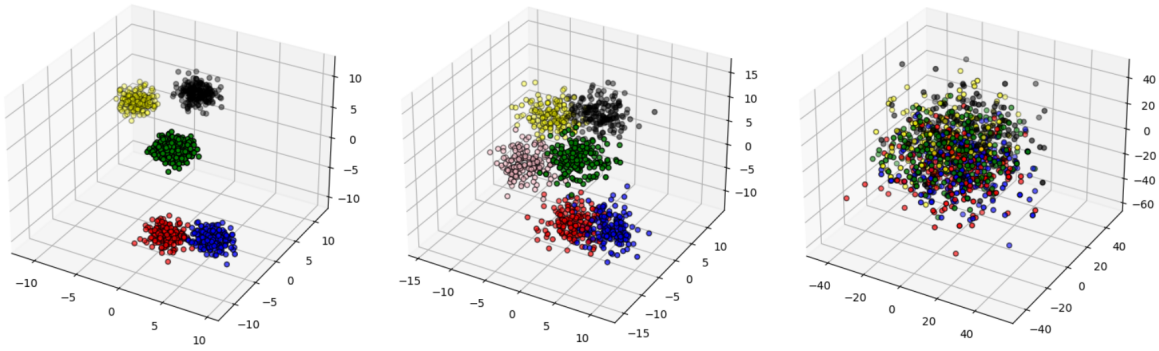
espalham em torno do centro.

Se o desvio padrão for **baixo**, os pontos de dados de cada *cluster* estarão mais concentrados em torno do centro; isso resulta em *clusters* mais densos e bem definidos, com menos sobreposição entre um grupo e outro, facilitando, também, na identificação destes por algoritmos de agrupamento (clustering).

Já se o desvio padrão for **alto**, os pontos de dados estarão mais dispersos em torno do centro de cada *cluster*. Isso pode resultar em *clusters* mais espalhados e menos distintos, com maior sobreposição entre si. Algoritmos de *clustering* podem ter mais dificuldade em distinguir grupos sobrepostos ou próximos.

Na Figura 10 é possível observar que, com um desvio padrão de 1, os grupos estão bem definidos, facilitando na visualização do *dataset*. Já quando aumenta-se para 7 a 15, observa-se que os pontos estão mais dispersos e já não é mais possível identificar os grupos com clareza.

Figura 10 – Variando o Desvio Padrão dos Clusters



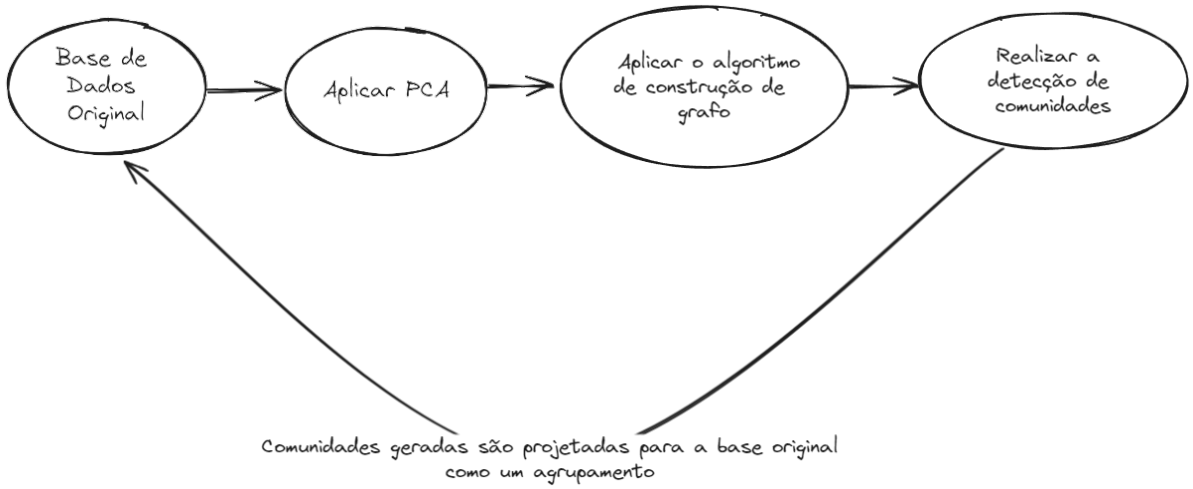
(a) Desvio Padrão: 1 **(b)** Desvio Padrão: 7 **(c)** Desvio Padrão: 15

Fonte: desenvolvido pela autora

3.2 Aplicando a Redução de Dimensionalidade

Antes de partir para os experimentos e consecutivos resultados, é necessário verificar o funcionamento do algoritmo PCA para a redução de dimensionalidade. Para isso, foi desenvolvido um fluxograma para organizar e especificar os processos que serão realizados. A Figura 11 descreve as tarefas realizadas e, nas Figuras 12 e 13, os resultados obtidos em cada etapa.

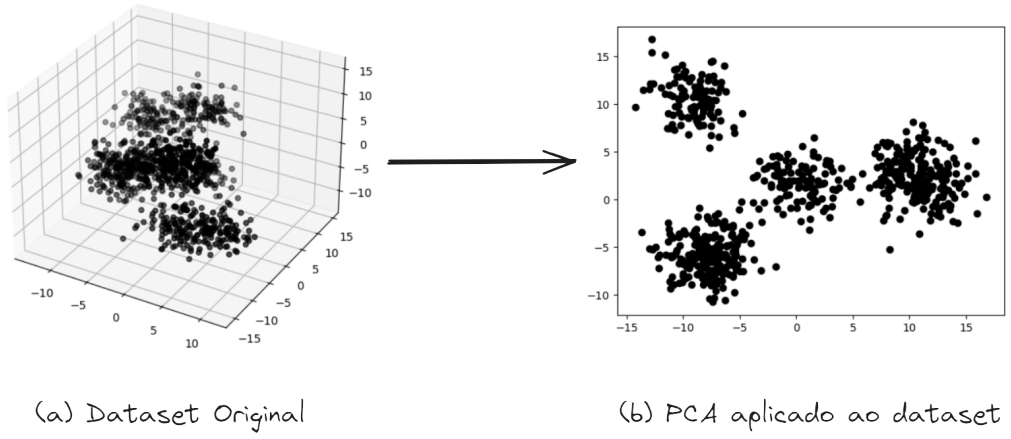
Figura 11 – Fluxograma de tarefas



Fonte: desenvolvido pela autora

As duas primeiras etapas consistem em tratar os dados gerados pelo dataset original (etapa 1) utilizando a redução de dimensionalidade com o algoritmo PCA (etapa 2). Com isso, reduziu-se os dados a 2 componentes principais, como é possível observar na Figura 12

Figura 12 – Tratamento de dados com alta dimensionalidade usando o PCA

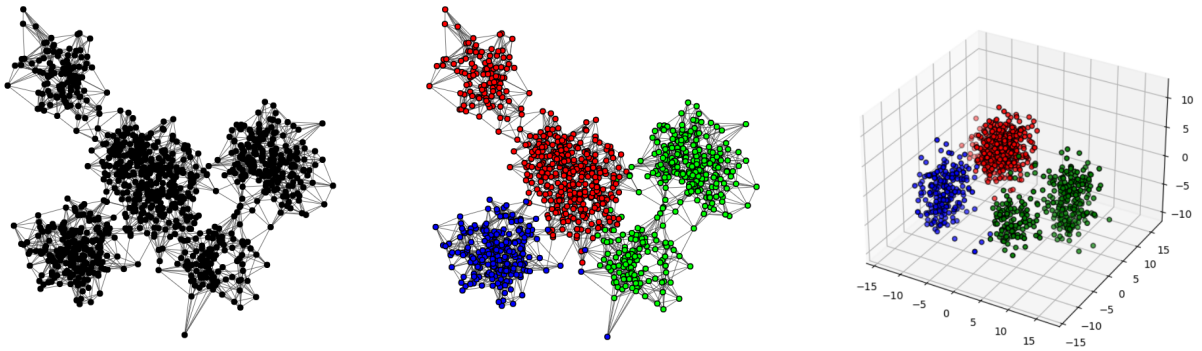


Fonte: desenvolvido pela autora

A partir disso, é possível aplicar o algoritmo *k-NN* para a construção do grafo, em seguida realizar a detecção de comunidades com o algoritmo *fastgreedy* e, por fim,

as comunidades são projetadas como um agrupamento nos dados originais. Assim, como demonstrado na Figura 13c, o agrupamento final irá consistir de apenas 3 grupos principais.

Figura 13 – Resultado Final



(a) Algoritmo k-NN aplicado **(b)** Resultado Final **(c)** Visualização em 3D

Fonte: desenvolvido pela autora

Sabendo do funcionamento das variáveis na criação dos datasets e também da aplicação da redução de dimensionalidade, será possível realizar variações nos parâmetros e simular o comportamento dos algoritmos em cenários próximos ao mundo real. Portanto, a Tabela 1 descreve quais intervalos cada parâmetro irá variar e qual seu respectivo incremento.

Tabela 1 – Intervalo de variação dos parâmetros para a geração dos conjuntos de dados.

Parâmetro	**Intervalo**	**Incremento**
Quantidade de classes	3 - 1000	5
Quantidade de amostras	100 - 1001	1
Quantidade de grupos	1 - 51	1
Sobreposição	10 - 50	1

3.3 Algoritmos Utilizados

Os algoritmos escolhidos para realizar os experimentos foram especificados no Capítulo 2. Em suma, k-NN, mk-NN, Kmeans, Agrupamento Aglomerativo e Fastgreedy. Nos tópicos a seguir, alguns parâmetros foram fixados para realizar os testes e verificar o desempenho dos algoritmos:

***K-means*:** [scikit-learn] - class sklearn.cluster.Kmeans()

- *init*: k-means++
- *n_init*: 10
- *max_iter*: 300

- *algorithm*: Lloyd (LLOYD, 1982)

***Agglomerative Clustering*:** Implementação do scikit-learn

- *linkage*: Ward
- *compute_distances*: falso

k-NN e mk-NN [scikit-learn]: kneighbors_graph()

- *mode*: distance
- *metric*: euclidean
- *include_self*: falso

Fastgreedy: Implementação do *igraph*

- *weights*: Peso da rede ponderada

3.4 Construção dos grafos

Esta pesquisa visa observar o desempenho de algoritmos de agrupamento com detecção de comunidades utilizando a estrutura de grafos. Para isso, focou-se nos algoritmos citados no Capitulo 3: k-NN e Mk-NN.

3.5 Experimentos

Os experimentos foram executados utilizando a linguagem Python, bem como as bibliotecas mencionadas no início deste capítulo. Assim, o código dos testes foi divido em:

1. Os parâmetros para criação do dataset foram escolhidos um a um e, para cada teste, utilizou-se o incremento definido na Tabela 1;
2. cada algoritmo foi executado dez vezes (iniciando em 1 e incrementando + 1 até chegar em 10) e utilizou-se de dados distintos para cada iteração, mantendo o valor dos parâmetros que não seriam analisados e variando somente aquele em que o teste seria feito;
3. em cada uma das dez iterações calculou-se a média e o desvio padrão do desempenho dos algoritmos, salvando os valores para realizar a análise no final da execução da iteração;

4. os valores de k para os algoritmos k-NN e mk-NN foram estabelecidos previamente pelo usuário, assim como mostra a Tabela 2. Somente a execução em que k obteve o desempenho máximo - ao comparar-se com as demais execuções - foi considerada para a análise.

Tabela 2 – Intervalo de k escolhido pelo usuário.

Algoritmo	**Parâmetro**	**Intervalo**	**Incremento**
k-NN	k	1 - 40	1
Mk-NN	k	1 - 40	1

Assim, dado o efeito que os algoritmos de construção de grafos tem no desempenho final gerado pelos algoritmos de Detecção de Comunidades todas as combinações citadas foram realizadas, sendo escolhido o melhor valor da métrica dado os diferentes valores dos parâmetros.

3.6 Testes estatísticos

Existem várias abordagens estatísticas que podem ser utilizadas para realizar análises comparativas do desempenho de algoritmos de agrupamento em diferentes conjuntos de dados, mas escolher a mais adequada requer experiência para discernir se as diferenças observadas nos resultados dos algoritmos são reais ou apenas uma coincidência.

Por isso, realizou-se uma avaliação estatística seguindo as recomendações de Demšar, que incluem, por exemplo, o uso de testes estatísticos como o ANOVA, teste t-pareado e teste de Friedman, para comparar diferentes algoritmos em um mesmo conjunto de dados. (DEMŠAR, 2006).

Para facilitar esse processo, foi empregado o pacote *Autorank* no Python (HERBOLD, 2020), que tem como objetivo tornar a análise estatística mais acessível para pessoas sem especialização na área. O *Autorank* simplifica o processo ao resumir todas as etapas recomendadas por Demšar em um único comando e oferece recursos adicionais para criar representações gráficas e tabelas de resultados. Para isso, os dados devem estar organizados em *dataframes* da biblioteca Pandas[4] do Python.

O processo utilizado pelo *Autorank* inclui, inicialmente, a aplicação do teste de Shapiro-Wilk para verificar se os resultados seguem uma distribuição normal (utilizando a Correção de Bonferroni para validar esses testes). Caso esta afirmativa seja verdadeira, o teste de Bartlett será usado para avaliar a homogeneidade; se for negativa, o teste de Levene é aplicado.

[4] <https://pandas.pydata.org/>

Com isso, o *Autorank* determina os testes e técnicas mais apropriados para estabelecer os intervalos de confiança usados na comparação estatística a partir dos resultados dos testes de normalidade e homogeneidade.

4 Análise e discussão dos resultados

Este capítulo é dedicado à apresentação dos resultados obtidos através da realização dos experimentos descritos no Capítulo 3. Para uma melhor organização e compreensão, este capítulo está estruturado em 4 seções distintas, explorando:

1. Variação da quantidade de atributos (*features*) para mensurar a qualidade dos grafos;
2. Variação da quantidade de grupos (*clusters*);
3. Variação da sobreposição (*std*).
4. Variação da quantidade de amostras;

4.1 Variando a quantidade de atributos (*features*) para mensurar a qualidade dos grafos

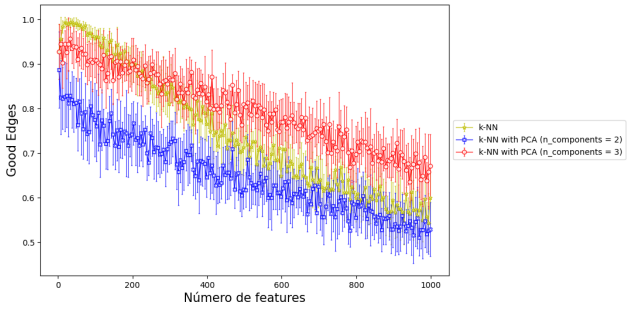
A Figura 14c mostra a qualidade, considerando a métrica *Good Edges*, dos grafos gerados pelo algoritmo k-NN sem aplicação de redução de dimensionalidade e k-NN com aplicação de PCA, variando o número de componentes para 2 e 3. O mesmo se repete na Figura 14d, porém utilizando o algoritmo mk-NN e suas respectivas variações utilizando o PCA.

Para a criação desse experimento foram utilizados os parâmetros:

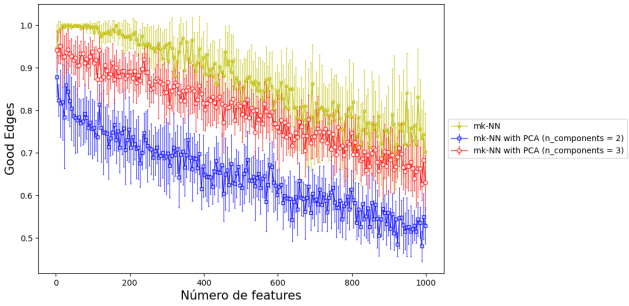
- *amostras*: 100
- *features*: Variou de 3 até 1000, incrementando de 5 em 5
- *std*: Variou a raiz quadrada de 3 até 1000, incrementando de 5 em 5
- *clusters*: 6

Foi preciso alterar tanto o número de atributos quanto o desvio padrão dos grupos, considerando que o método *make blobs* gera um conjunto de dados sintéticos. Isso visa imitar dados reais, que normalmente se tornam mais dispersos à medida que novos atributos são adicionados. A Tabela 3 foi elaborada para apresentar os resultados do desempenho médio e do desvio padrão da métrica utilizada, em relação às arestas identificadas por cada algoritmo.

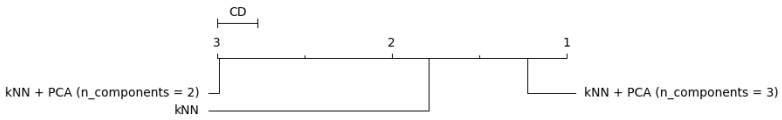
Figura 14 – Variando a quantidade de *features*



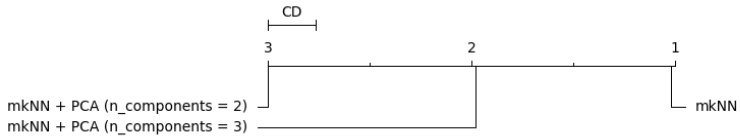
(a) Qualidade dos grafos - k-NN



(b) Qualidade dos grafos - mk-NN



(c) Comparação estatística - k-NN



(d) Comparação estatística - mk-NN

Fonte: desenvolvido pela autora

Tabela 3 – Good Edges Médio e Desvio Padrão variando o número de atributos.

Algoritmo de construção de grafos	**Média**	**Desvio Padrão**
k-NN	0.822289341195216	0.12363631652289686
k-NN with PCA (n_components = 2)	0.6972543172514287	0.07806837288693132
k-NN with PCA (n_components = 3)	0.8458605904027352	0.06874625603066901
mk-NN	0.921136426068305	0.07539994970107414
mk-NN with PCA (n_components = 2)	0.6950528716339811	0.0767985181065246
mk-NN with PCA (n_components = 3)	0.844748091432415	0.06934934042495429

A análise estatística é representada pelas Figuras 14c e 14d e foi realizada para 3 populações com 200 amostras pareadas. O nível de significância familiar dos testes é alfa=0,050. Rejeitou-se a hipótese nula de que a população é normal para as populações

k-NN + PCA (n_components = 2), k-NN, k-NN + PCA (n_components = 3), mk-NN + PCA (n_components = 2), mk-NN + PCA (n_components = 3) e mk-NN. Portanto, assumiu-se que nem todas as populações são normais.

Como tem-se mais de duas populações e algumas delas não são normais, o *autorank* escolheu utilizar o teste não paramétrico de Friedman como teste omnibus para determinar se existem diferenças significativas entre os valores médios das populações. Também utilizou-se o teste post-hoc de Nemenyi para inferir quais diferenças são significativas.

Diferenças entre populações são significativas se a diferença da classificação média for maior do que a distância crítica CD=0,234 do teste de Nemenyi. Para o teste de Friedman, rejeitou-se a hipótese de que não há diferença na tendência central das populações. Portanto, assumiu-se que há uma diferença estatisticamente significativa entre os valores médios das populações.

Com base nos resultados do teste post-hoc de Nemenyi, o *autorank* determinou que todas as diferenças entre as populações são significativas.

Comparando os resultados obtidos na Tabela 3 e Figuras 14c e 14d, conclui-se que os algoritmos para criação de grafos utilizando mk-NN e k-NN + PCA (n_components = 3) foram mais eficientes, enquanto que os algoritmos k-NN + PCA (n_components = 2) e mk-NN + PCA (n_components = 2) ocupam o último lugar em questão de eficiência. Presume-se que a quantidade de componentes igual a 2 ocasionou a perda de informações durante a criação dos grafos.

Com esse experimento, foi possível compreender a interferência da quantidade de *features* e consequente efeito da Maldição da Dimensionalidade. Percebe-se que, ao aumentar significativamente a quantidade de atributos, a qualidade dos grafos diminui 2.4.2.

4.2 Variação da quantidade de grupos (*clusters*)

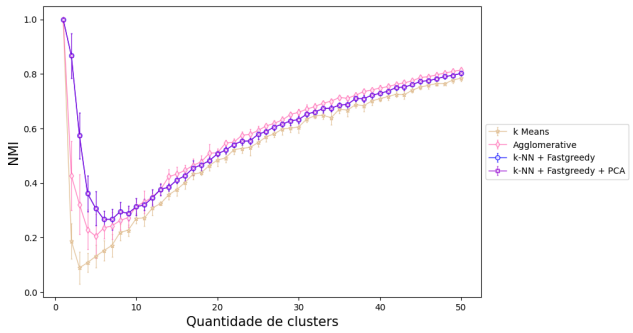
Para a criação desse experimento foram utilizados os parâmetros:

- *amostras*: 100
- *features*: 1000
- *std*: 42
- *clusters*: variando de 1 a 51, incremento de +1 por iteração

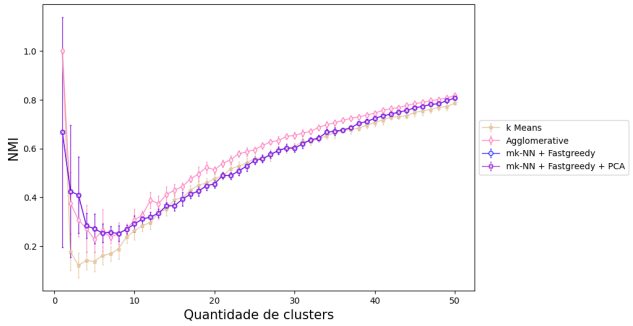
O número de componentes para a aplicação do PCA foi fixado em 3. As Figuras 15a e 15b demonstra o desempenho dos algoritmos tradicionais e de grafos com detecção

de comunidade variando a quantidade de grupos (*clusters*). A Tabela 4 foi criada com os resultados da média e desvio padrão da métrica NMI alcançados por cada algoritmo dada a variação.

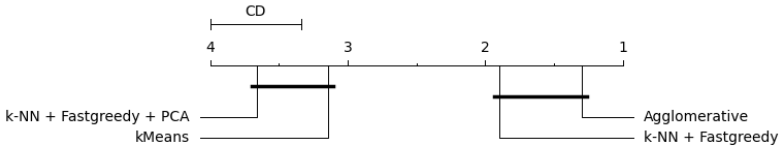
Figura 15 – Variando a quantidade de *clusters*



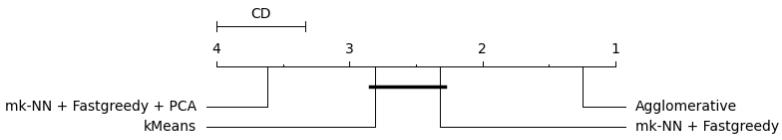
(a) Variando a quantidade de Clusters - k-NN



(b) Variando a quantidade de Clusters - mk-NN



(c) Comparação estatística - k-NN



(d) Comparação estatística - mk-NN

Fonte: desenvolvido pela autora

Tabela 4 – NMI Médio e Desvio Padrão variando a quantidade de amostras.

Algoritmo de agrupamento	**Média**	**Desvio Padrão**
K-means	0.4226522497567831	0.22638877088309697
Agglomerative	0.4860612643073749	0.1988574284914012
k-NN + Fastgreedy	0.5134811782176172	0.19137070897711836
k-NN + Fastgreedy + PCA	0.33438988872338726	0.14897583782041615
Mk-NN + Fastgreedy	0.45087129959865474	0.15538497976434787
Mk-NN + Fastgreedy + PCA	0.3131296744376575	0.1269134657456902

A análise estatística é representada pelas Figuras 15c e 15d e foi realizada para 4 populações com 50 amostras pareadas. O nível de significância familiar dos testes é alfa=0,050. Assumiu-se que nem todas as populações são normais. Como tem-se mais de duas populações e algumas delas não são normais, o *autorank* escolheu utilizar o teste não paramétrico de Friedman como teste omnibus para determinar se existem diferenças significativas entre os valores médios das populações. Também utilizou-se o teste post-hoc de Nemenyi para inferir quais diferenças são significativas.

Diferenças entre populações são significativas se a diferença da classificação média for maior do que a distância crítica CD=0,663 do teste de Nemenyi. Para o teste de Friedman, rejeitou-se a hipótese de que não há diferença na tendência central das populações.

Com base nos resultados do teste post-hoc de Nemenyi, o *autorank* determinou que não há diferença significativa entre os grupos de algoritmos (kMeans, mk-NN + Fastgreedy e k-NN + Fastgreedy + PCA), (k-NN + Fastgreedy e Agglomerative). Todas as outras diferenças entre as populações são significativas.

Comparando os resultados obtidos na Tabela 4 e Figuras 15c e 15d, conclui-se que os algoritmos para criação de grafos utilizando k-NN + Fastgreedy foram mais eficientes, enquanto que os algoritmos Mk-NN + Fastgreedy + PCA ocupam o último lugar.

4.3 Variação da sobreposição

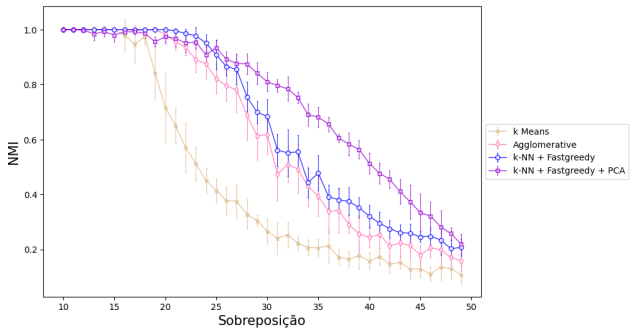
Para a criação desse experimento foi utilizado os parâmetros:

- *amostras*: 100
- *features*: 1000
- *std*: Variou de 10 até 50 com incremento de 1
- *clusters*: 6

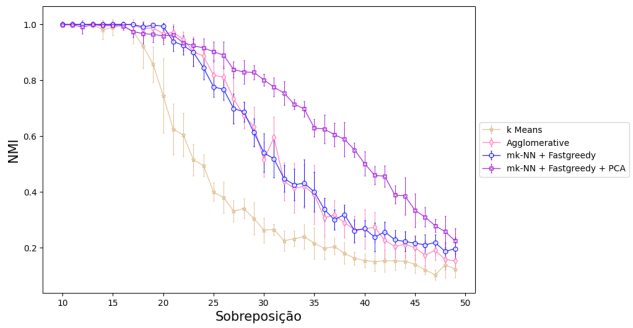
As Figuras 16a e 16b demonstram o desempenho dos algoritmos tradicionais e de grafos com detecção de comunidade variando a sobreposição (desvio padrão dos grupos),

utilizando o NMI como métrica. A Tabela 5 foi criada para agrupar os resultados da média e desvio padrão da métrica NMI alcançados por cada algoritmo dada a variação da sobreposição.

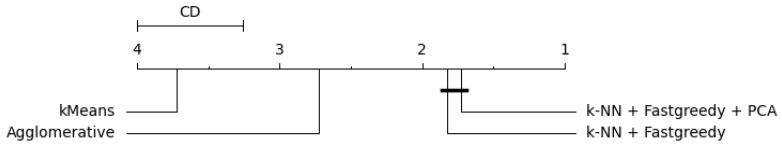
Figura 16 – Variando o desvio padrão do *dataset*



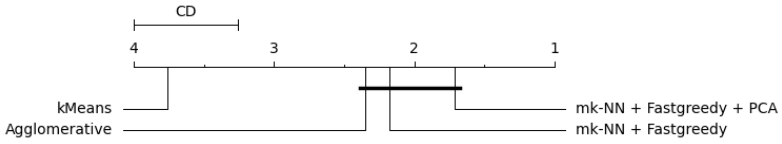
(a) Variando o desvio padrão - k-NN



(b) Variando a quantidade de Clusters - mk-NN



(c) Comparação estatística - k-NN



(d) Comparação estatística - mk-NN

Fonte: desenvolvido pela autora

Tabela 5 – NMI Médio e Desvio Padrão variando o desvio padrão dos dados.

Algoritmo de agrupamento	**Média**	**Desvio Padrão**
K-means	0.6265858147405803	0.3343610114369317
Agglomerative	0.7884902630586111	0.27181089331304253
k-NN + Fastgreedy	0.8278239919399076	0.2479706931053945
k-NN + Fastgreedy + PCA	0.7405667793119727	0.2479706931053945
Mk-NN + Fastgreedy	0.784157962694472	0.2690073084638369
Mk-NN + Fastgreedy + PCA	0.7303164054726407	0.2690073084638369

A análise estatística é representada pelas Figuras 16c e 16d e foi realizada para 4 populações com 40 amostras pareadas. O nível de significância familiar dos testes é alfa=0,050. Assumiu-se que nem todas as populações são normais. Como tem-se mais de duas populações e algumas delas não são normais, o *autorank* escolheu utilizar o teste não paramétrico de Friedman como teste omnibus para determinar se existem diferenças significativas entre os valores médios das populações. Também utilizou-se o teste post-hoc de Nemenyi para inferir quais diferenças são significativas.

Diferenças entre populações são significativas se a diferença da classificação média for maior do que a distância crítica CD=0,742 do teste de Nemenyi. Para o teste de Friedman, rejeitou-se a hipótese de que não há diferença na tendência central das populações.

Com base nos resultados do teste post-hoc de Nemenyi, o *autorank* determinou que não há diferença significativa entre os grupos de algoritmos (k-NN + Fastgreedy e k-NN + Fastgreedy + PCA), (Agglomerative, mk-NN + Fastgreedy e mk-NN + Fastgreedy + PCA). Todas as outras diferenças entre as populações são significativas.

Comparando os resultados obtidos na Tabela 5 e Figuras 16c e 16d, conclui-se que os algoritmos para criação de grafos utilizando k-NN + Fastgreedy foi mais eficientes, enquanto que o algoritmo K-means ocupa o último lugar.

4.4 Variando a quantidade de amostras

Para a criação desse experimento foram utilizados diversos tipos de parâmetros e realizou-se diferentes experimentos - visando observar como os algoritmos se comportam ao reduzir a dimensionalidade. Os testes foram divididos em:

- Teste 1:
  - *amostras*: Varia de 100 até 1001, incrementando de 100 em 100
  - *features*: 300
  - *std*: 42
  - *clusters*: 6

- Teste 2:
  - *amostras*: Varia de 100 até 1001, incrementando de 100 em 100
  - *features*: 650
  - *std*: 50
  - *clusters*: 6

4.4.1 Teste 1

As Figuras 17a e 17b demonstram o desempenho dos algoritmos tradicionais e de grafos com detecção de comunidade variando a quantiade de amostras, utilizando o NMI como métrica. A Tabela 6 foi criada para agrupar os resultados da média e desvio padrão da métrica NMI alcançados por cada algoritmo dada a variação da quantidade de amostras.

Tabela 6 – NMI Médio e Desvio Padrão variando a quantiade de amostras - Teste 1

Algoritmo de agrupamento	**Média**	**Desvio Padrão**
K-means	0.07371212399904419	0.018156100573875927
Agglomerative	0.07570755626169193	0.030606145950852995
k-NN + Fastgreedy	0.10652054721103552	0.03145118756504548
k-NN + Fastgreedy + PCA	0.16632919276705513	0.02413140247354827
Mk-NN + Fastgreedy	0.04583707677980003	0.058434073763113284
Mk-NN + Fastgreedy + PCA	0.1632517893366364	0.024610649836263262

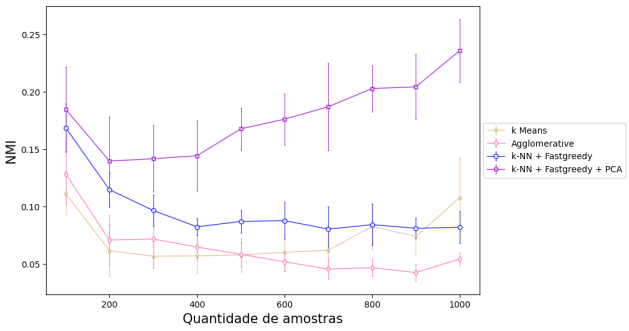
A análise estatística é representada pelas Figuras 17c e 17d e foi realizada para 4 populações com 10 amostras pareadas. O nível de significância familiar dos testes é alfa=0,050. Assumiu-se que nem todas as populações são normais. Como tem-se mais de duas populações e algumas delas não são normais, o *autorank* escolheu utilizar o teste não paramétrico de Friedman como teste omnibus para determinar se existem diferenças significativas entre os valores médios das populações. Também utilizou-se o teste post-hoc de Nemenyi para inferir quais diferenças são significativas.

Diferenças entre populações são significativas se a diferença da classificação média for maior do que a distância crítica CD=1,483 do teste de Nemenyi. Para o teste de Friedman, rejeitou-se a hipótese de que não há diferença na tendência central das populações.

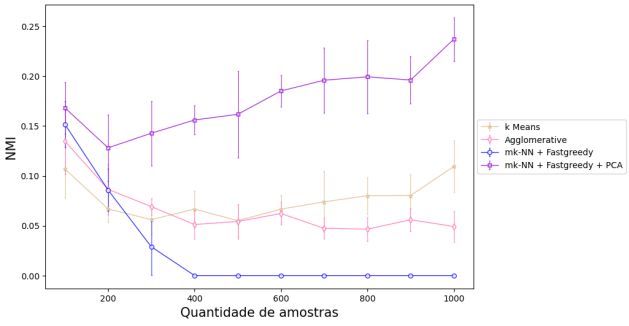
Com base nos resultados do teste post-hoc de Nemenyi, o *autorank* determinou que não há diferença significativa entre os grupos de algoritmos (mk-NN + Fastgreedy, kMeans, Agglomerative), (Agglomerative, k-NN + Fastgreedy e mk-NN + Fastgreedy + PCA). Todas as outras diferenças entre as populações são significativas.

Comparando os resultados obtidos na Tabela 6 e Figuras 17c e 17d, conclui-se que os algoritmos para criação de grafos utilizando a redução de dimensionalidade foram mais eficientes, enquanto que os algoritmos que não utilizam a técnica ocupam o último lugar.

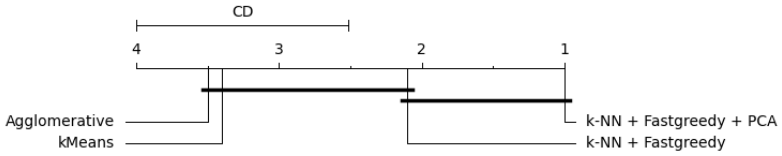
Figura 17 – Variando a quantidade de amostras - Teste 1



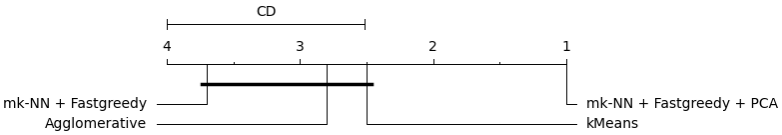
(a) Variando a quantidade de amostras - k-NN



(b) Variando a quantidade de amostras - mk-NN



(c) Comparação estatística - k-NN



(d) Comparação estatística - mk-NN

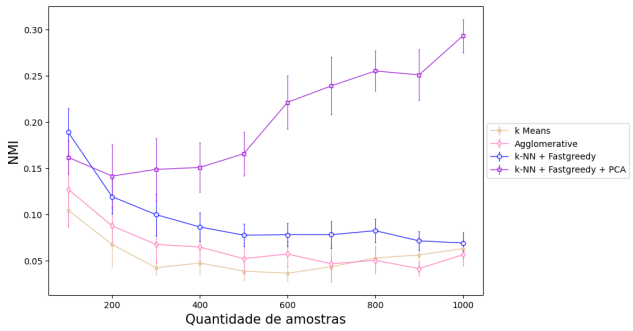
Fonte: desenvolvido pela autora

4.4.2 Teste 2

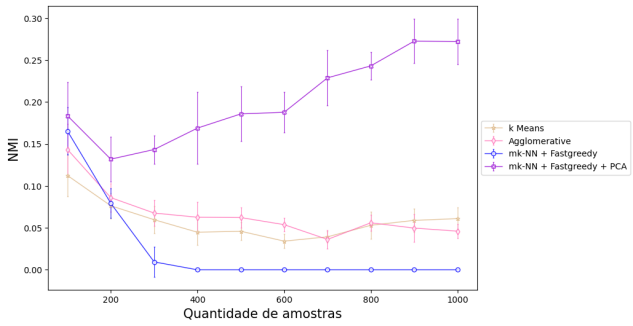
As Figuras 18a e 18b demonstram o desempenho dos algoritmos tradicionais e de grafos com detecção de comunidade variando a quantiade de amostras, utilizando o NMI como métrica. A Tabela 7 foi criada para agrupar os resultados da média e desvio padrão da métrica NMI alcançados por cada algoritmo dada a variação da quantidade de amostras.

A análise estatística é representada pelas Figuras 18c e 18d e foi realizada para 4 populações com 10 amostras pareadas. O nível de significância familiar dos testes é alfa=0,050. Assumiu-se que nem todas as populações são normais. Como tem-se mais de

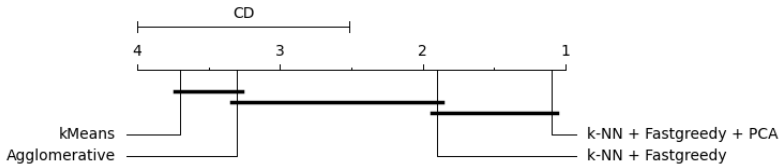
Figura 18 – Variando a quantidade de amostras - Teste 2



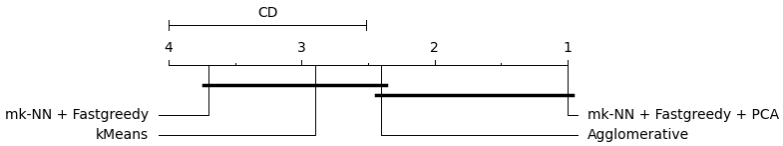
(a) Variando a quantidade de amostras - k-NN



(b) Variando a quantidade de amostras - mk-NN



(c) Comparação estatística - k-NN



(d) Comparação estatística - mk-NN

Fonte: desenvolvido pela autora

Tabela 7 – NMI Médio e Desvio Padrão variando a quantidade de amostras - Teste 2

Algoritmo de agrupamento	**Média**	**Desvio Padrão**
K-means	0.05917498951787773	0.023652686428685794
Agglomerative	0.07524857193260834	0.027749567603758893
k-NN + Fastgreedy	0.10913538325430017	0.04050351992712473
k-NN + Fastgreedy + PCA	0.1775456821535021	0.04129001362412219
Mk-NN + Fastgreedy	0.04443273354375815	0.06360126440882832
Mk-NN + Fastgreedy + PCA	0.1794382172595493	0.038186733106916525

duas populações e algumas delas não são normais, o *autorank* escolheu utilizar o teste não paramétrico de Friedman como teste omnibus para determinar se existem diferenças significativas entre os valores médios das populações. Também utilizou-se o teste post-hoc de Nemenyi para inferir quais diferenças são significativas.

Diferenças entre populações são significativas se a diferença da classificação média for maior do que a distância crítica CD=1,483 do teste de Nemenyi. Para o teste de Friedman, rejeitou-se a hipótese de que não há diferença na tendência central das populações.

Com base nos resultados do teste post-hoc de Nemenyi, o *autorank* determinou que não há diferença significativa entre os grupos de algoritmos (mk-NN + Fastgreedy, kMeans, Agglomerative), (Agglomerative, k-NN + Fastgreedy e mk-NN + Fastgreedy + PCA). Todas as outras diferenças entre as populações são significativas.

Comparando os resultados obtidos na Tabela 7 e Figuras 18c e 18d, conclui-se que os algoritmos para criação de grafos utilizando a redução de dimensionalidade foram mais eficientes, enquanto que os algoritmos que não utilizam a técnica ocupam o último lugar.

5 Conclusão

Neste trabalho, explorou-se em profundidade o campo de agrupamento em conjuntos de dados com múltiplas variáveis, utilizando técnicas de redução de dimensionalidade e detecção de comunidades. Através dos experimentos realizados, foi possível avaliar a eficácia de diferentes abordagens algorítmicas em *datasets* complexos, com um foco na variabilidade dos números de amostras, grupos e desvios padrão em conjuntos de dados com um elevado número de características.

Os resultados obtidos demonstraram claramente que os algoritmos que combinam k-nn ou m-nn com o método *fastgreedy*, seguido da redução de dimensionalidade via PCA, se destacaram em termos de desempenho. Esta combinação mostrou-se eficaz na destilação de informações relevantes de grandes conjuntos de dados, permitindo uma melhor visualização e interpretação dos agrupamentos formados.

A eficiência desses métodos, especialmente em contextos onde o número de amostras é elevado e as variáveis são múltiplas e complexas, ressalta a importância de técnicas sofisticadas de análise de dados. A redução de dimensionalidade via PCA provou ser uma ferramenta valiosa não apenas para simplificar a complexidade dos dados, mas também para realçar estruturas significativas que poderiam ser obscurecidas em dimensões mais elevadas.

Em resumo, este trabalho contribui para o campo de agrupamento de dados, oferecendo *insights* significativos e analisando a eficácia de combinar técnicas de agrupamento baseadas em vizinhança com métodos de redução de dimensionalidade e detecção de comunidades. Essas descobertas abrem caminho para futuras pesquisas que podem explorar outras variantes dessas técnicas ou aplicá-las em diferentes domínios de dados, potencialmente revelando novas perspectivas e soluções inovadoras para desafios complexos na análise de dados.

Referências

BRITO, M. et al. Connectivity of the mutual k-nearest-neighbor graph in clustering and outlier detection. *Statistics Probability Letters*, v. 35, n. 1, p. 33–42, 1997. ISSN 0167-7152. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167715296002131>. Citado na página 29.

BROWN, S. *Machine learning, explained*. 2021. <https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained>. Citado 2 vezes nas páginas 19 e 23.

CLAUSET, A.; NEWMAN, M. E. J.; MOORE, C. Finding community structure in very large networks. *Phys. Rev. E*, American Physical Society, v. 70, p. 066111, Dec 2004. Disponível em: <https://link.aps.org/doi/10.1103/PhysRevE.70.066111>. Citado na página 30.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, v. 7, p. 1–30, 2006. Citado na página 41.

DINH, T. N. Graph data structure cheat sheet for coding interviews. *Towards Data Science*, 2020. Disponível em: <https://towardsdatascience.com/graph-data-structure-cheat-sheet-for-coding-interviews-a38aadf8aa87>. Citado na página 27.

DRIDI, S. Supervised learning-a systematic literature review. OSF Preprints, 2021. Disponível em: <https://files.osf.io/v1/resources/tysr4/providers/osfstorage/624a442a7a7d9e04500608ce?action=download&direct&version=3>. Citado na página 24.

FEOFILOFF, P. *Grafos*. [s.n.], 2017. Disponível em: <https://www.ime.usp.br/~pf/algoritmos_para_grafos/aulas/graphs.html#:~:text=Grafos%20n%C3%A3o%2Ddirigidos,define%20um%20grafo%20n%C3%A3o%2Ddirigido.> Citado na página 27.

FORTUNATO, S. Community detection in graphs. *Physics Reports*, v. 486, n. 3, p. 75–174, 2010. ISSN 0370-1573. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0370157309002841>. Citado na página 30.

GUO, G. et al. Knn model-based approach in classification. In: SPRINGER. *OTM Confederated International Conferences"On the Move to Meaningful Internet Systems"*. [S.l.], 2003. p. 986–996. Citado na página 28.

HARTIGAN, J. A. Statistical theory in clustering. *Journal of Classification*, v. 2, n. 1, p. 63–76, 12 1985. ISSN 1432-1343. Disponível em: <https://doi.org/10.1007/BF01908064>. Citado na página 31.

HASAN, B. M. S.; ABDULAZEEZ, A. M. A review of principal component analysis algorithm for dimensionality reduction. *Journal of Soft Computing and Data Mining*, v. 2, n. 1, p. 20–30, 2021. Citado na página 34.

HERBOLD, S. Autorank: A python package for automated ranking of classifiers. *Journal of Open Source Software*, The Open Journal, v. 5, n. 48, p. 2173, 2020. Disponível em: <https://doi.org/10.21105/joss.02173>. Citado na página 41.

JAIN, A. K. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, v. 31, n. 8, p. 651–666, 2010. ISSN 0167-8655. Award winning papers from the 19th International Conference on Pattern Recognition (ICPR). Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167865509002323>. Citado na página 25.

JAIN, A. K.; DUBES, R. C. *Algorithms for clustering data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988. Disponível em: <http://portal.acm.org/citation.cfm?id=46712>. Citado na página 31.

JIA, W. et al. Feature dimensionality reduction: a review. *Complex & Intelligent Systems*, v. 8, n. 3, p. 2663–2693, 06 2022. ISSN 2198-6053. Disponível em: <https://doi.org/10.1007/s40747-021-00637-x>. Citado na página 33.

JONES, M. T. Unsupervised learning for data classification. IBM Developer, December 2017. Disponível em: <https://developer.ibm.com/articles/cc-unsupervised-learning-data-classification/?mhsrc=ibmsearch_a&mhq=%20unsupervised>. Citado na página 24.

JUSTEL, C. et al. Detecção de comunidades numa rede de relacionamento de alunos. *Sistemas & Gestão*, v. 9, n. 4, p. 480–487, 2014. Citado na página 30.

LLOYD, S. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, v. 28, n. 2, p. 129–137, 1982. Citado na página 40.

MAJEED, A.; RAUF, I. Graph theory: A comprehensive survey about graph theory applications in computer science and social networks. *Inventions*, v. 5, n. 1, 2020. ISSN 2411-5134. Disponível em: <https://www.mdpi.com/2411-5134/5/1/10>. Citado na página 28.

MITCHELL, T. M. *Machine Learning*. [S.l.]: McGraw-Hill Education(ISE Editions), 1997. Citado na página 19.

MONTEIRO, R. B. Comparação de técnicas de aprendizado de máquina para predição da disponibilidade de bicicletas no projeto bicicletar fortaleza. 2018. Citado na página 23.

OZAKI, K. et al. Using the mutual k-nearest neighbor graphs for semi-supervised classification on natural language data. In: *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*. Portland, Oregon, USA: Association for Computational Linguistics, 2011. p. 154–162. Disponível em: <https://aclanthology.org/W11-0318>. Citado na página 32.

SINAGA, K. P. Unsupervised k-means clustering algorithm. *IEEE Access*, 2020. Disponível em: <https://ieeexplore.ieee.org/ielx7/6287639/8948470/09072123.pdf>. Citado 3 vezes nas páginas 19, 25 e 26.

SIRIMONGKOLKASEM, T.; DRIKVANDI, R. On regularisation methods for analysis of high dimensional data. *Annals of Data Science*, v. 6, n. 4, p. 737–763, 12 2019. ISSN 2198-5812. Disponível em: <https://doi.org/10.1007/s40745-019-00209-4>. Citado na página 32.

STREHL, A.; GHOSH, J. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, v. 3, n. Dec, p. 583–617, 2002. Citado na página 31.

VLACHOS, M. Dimensionality reduction. In: _____. *Encyclopedia of Machine Learning*. Boston, MA: Springer US, 2010. p. 274–279. ISBN 978-0-387-30164-8. Disponível em: <https://doi.org/10.1007/978-0-387-30164-8_216>. Citado na página 32.

ZHU, X. *Semi-supervised learning literature survey*. [S.l.], 2005. Citado na página 27.