Comparação de Desempenho entre Algoritmos de Agrupamentos

Benchmarking Clustering Algorithms

Leonardo Rafael Bueno* Rafael Gomes Mantovani*

RESUMO

Atualmente existe uma crescente procura e interesse na implantação de Inteligência Artificial e Aprendizado de Máquina em diferentes áreas de aplicação, como sistemas de recomendação, gerenciamento de rotas, controle de spams, entre outros. Muitas dessas tarefas podem ser modeladas por meio de diferentes algoritmos, sejam eles de aprendizado supervisionado ou não supervisionados. Os algoritmos não supervisionados, em específico, tendem a trabalhar com tarefas mais desafiadoras, uma vez que não se tem informação dos rótulos dos exemplos de um conjunto de dados. Além disso, como saber qual é o melhor algoritmo para um determinado problema? Infelizmente, não existe uma regra ou indicação que recomende os melhores algoritmos. Até então, a literatura tem se dedicado a investigar e avaliar diferentes algoritmos em um problema, e assim tentar escolher um que se adeque mais à tarefa em questão. Neste trabalho comparamos três algoritmos de agrupamento de dados em um problema real, no intuito de verificar características que possibilitem automatizar a escolha do algoritmo em futuros experimentos. No conjunto de dados estudado, todos os algoritmos apresentaram quase o mesmo valor de coeficiente de silhueta, entretanto, a maneira de lidar com cada algoritmo é completamente diferente.

Palavras-chave: Aprendizado de Máquina, Aprendizado Não supervisionado, Agrupamento de Dados

ABSTRACT

Currently, there is a growing demand and interest in the implementation of Artificial Intelligence and Machine Learning in different application areas, such as recommendation systems, route management, spam control, among others. Many of these tasks can be modeled using different algorithms, whether supervised or unsupervised learning. Unsupervised algorithms, in particular, tend to work with more challenging tasks, since there is no information on examples of examples of a data set. Also, how do you know the best algorithm for a given problem? Unfortunately, there is no rule or guideline that recommends the best algorithms. Until then, the literature has been dedicated to investigating and evaluating different algorithms in a problem, and thus trying to choose one that best suits the task at hand. In this work we compare three data clustering algorithms in a real problem, without the intention of features that make it possible to automate an algorithm choice in future experiments. In the studied dataset, all parallel algorithms have almost the same silhouette coefficient value, however, one way of dealing with each algorithm is completely different.

Keywords: Machine Learning, Unsupervised Learning, Clustering

1 INTRODUÇÃO

Atualmente existe uma crescente procura e interesse na implantação de Inteligência Artificial (IA) em todo âmbito da vida moderna (RUSSELL; NORVIG, 2010). Isso inclui, especificamente, o desenvolvimento

*Engenharia de Computação, Universidade Tecnológica Federal do Paraná, Campus Apucarana;
leonardorafaelbueno@alunos.utfpr.edu.br, rafaelmantovani@utfpr.edu.br

e uso de aplicações de Aprendizado de Máquina (AM) (MARSLAND, 2009), tanto no mercado de trabalho como em tarefas do dia a dia. É comum ver o uso de algoritmos inteligentes em: aplicativos de smartphones, na geração de rotas por GPS, indicações de anúncios, indicações de séries e filmes, em gerenciamentos de empresas, em dispositivos inteligentes como a Alexa e Siri, reconhecimento de spams, entre outros (RUSSELL; NORVIG, 2010). Muitas dessas tarefas podem ser modeladas por meio de diferentes algoritmos, que podem ser subdivididos em duas categorias, dependendo do tipo de dados disponível para a tarefa: aprendizado supervisionado e aprendizado não-supervisionado (MARSLAND, 2009).

O aprendizado supervisionado designa tarefas de manipulação de dados onde os mesmos possuem rótulo, ou seja, existe um atributo preditivo que especifica categorias ou valores numéricos para cada exemplo do *dataset* (conjunto de dados). Esses algoritmos usam essa informação já disponível no *dataset* para gerar conhecimento e aprender algo novo. Se o atributo preditivo foi categórico, denominamos a tarefa por *classificação*. Se o atributo preditivo for numérico, chamamos a tarefa de regressão (LUGER, 2013). Em ambos os casos, os algoritmos supervisionados usam essa informação presente nos dados (a classe) para guiar o processo de aprendizado mediante acertos e erros. Toda vez que algo for predito erroneamente o algoritmo se adapta para aprender aquele conceito e acertar as próximas predições.

Já o aprendizado não-supervisionado designa uma tarefa onde trabalha-se com dados não-rotulados, ou seja, temos apenas características descritivas de um problema, e nem ao menos sabemos se existem padrões entre essas instâncias (LUGER, 2013). Dessa forma, tenta-se extrair alguma informação ou significado dos conjuntos de dados, desconhecendo a priori o que será identificado. Algoritmos não-supervisionados tentam encontrar padrões ocultos, e estabelecer relações entre os dados, muitas vezes não tendo tanta clareza sobre o que será previsto.

Os algoritmos de aprendizado não-supervisionado também são comumente chamados de algoritmos de agrupamento de dados, ou *clustering* (RUSSELL; NORVIG, 2010). Além disso, tarefas de agrupamento tendem a ser mais desafiadoras, uma vez que não se tem informação dos rótulos dos exemplos. Outro ponto de dificuldade é: como saber qual é o melhor algoritmo para um determinado problema? Infelizmente, não existe uma regra ou indicação que recomende os melhores algoritmos. Até então, a literatura tem se dedicado a investigar e avaliar diferentes algoritmos em um problema, e assim tentar escolher um que se adeque mais à tarefa em questão.

Sendo assim, este trabalho tem por objetivo comparar e avaliar diferentes algoritmos de agrupamento de dados em uma mesma base de dados (*dataset*). Almeja-se com o processo extrair algum conhecimento que ajude a automatizar essa escolha em futuras execuções e experimentos. Os experimentos foram realizados com uma base de dados pública e três algoritmos de agrupamento de dados, a serem explicados com mais detalhes a seguir.

2 MÉTODO

2.1 Conjunto de dados

O conjunto de dados usado nos experimentos foi o "*Star Cluster Simulation*"[1], um *dataset* que representa informações de um aglomerado de estrelas. Alguns modelos populares de aglomerados são podem ser simulados a partir de N corpos, onde cada uma das estrelas é representada por meio de uma partícula que

[1] https://www.kaggle.com/mariopasquato/star-cluster-simulations/version/1

interage gravitacionalmente com as demais. O *dataset* contém a posição e velocidades de 64.000 estrelas simuladas e distribuídas no espaço de velocidade e posição. Existe um total de 8 colunas descrevendo as características de cada estrela. As colunas 1, 2, 3 representam respectivamente as posições x, y e z. Nas colunas 4, 5 e 6 existem as informações das velocidades de vx, vy e vz. A coluna 7 representa a massa de cada estrela. Na coluna 8 são representados os números de "id" (identificação) de cada estrela.

2.2 Algoritmos de Agrupamento de Dados

Como mencionado na introdução, os algoritmos de agrupamento de dados tem por objetivo unir os exemplos de um conjunto em grupos com conteúdos semelhantes. Assim, dados com informações diferentes tendem a formar grupos distintos gerando uma diferenciação do conteúdo em cada grupo. As principais etapas de qualquer algoritmo são: busca de similaridades, formação de conjuntos por diferenciação de classes e subclasses, e a formação de diferentes estruturas (grupos). Os algoritmos de agrupamento são também descritos como técnicas de mineração de dados (TORGO, 2011), pois são usados em etapas de análise exploratória de dados quando não se tem nenhuma informação sobre a "forma" dos dados.

2.2.1 Algoritmo de K-Médias (K-Means)

O primeiro algoritmo que estudamos foi o algoritmo de agrupamento das K-Médias (K-Means) (MARSLAND, 2009). O K-Means é um algoritmo cujo funcionamento principal é formar *k* conjuntos diferenciados baseado nos dados de entrada. Ele funciona com função de distância calculando a similaridade entre as instâncias. No início, são gerados k centróides, pontos aleatoriamente distribuídos dentro do espaço de representação do *dataset*. Esses pontos servem de referência para os grupos gerados. A cada iteração, uma instância é selecionada, e sua distância mensurada em relação a cada centróide. O exemplo então é associado ao centróide (grupo) de menor distância. Isso implica também em atualizar a nova posição do centróide, já que um novo elemento faz parte do grupo. O processo se repete até que todas as instâncias tenham sido organizadas nos grupos. A principal vantagem é que o K-Means é um dos algoritmos mais simples para agrupamentos de dados. Por outro lado, é preciso experimentar e testar vários valores de k para saber qual a melhor quantidade de grupos.

2.2.2 Agrupamento Hierárquico

O segundo algoritmo usado foi o Agrupamento Hierárquico (*Hierarchical Clustering*) (MITCHELL, 1997), um algoritmo que procura construir uma hierarquia de grupos. Existem duas estratégias para agrupamentos hierárquicos: a) aglomerativo - ou também chamada de cima para baixo, cada instância começa em um próprio cluster, e pares de clusters são mesclados à medida que se sobe na hierarquia; b) divisivo - ou de cima para baixo, todas as instâncias começam um cluster único, e as divisões são executadas recursivamente à medida que se desce a hierarquia. Em ambos os casos o algoritmo necessita de dois recursos: um critério de ligação e uma métrica de distância. Os resultados do agrupamento hierárquico são geralmente apresentados em um dendograma.

2.2.2 Agrupamento por Densidade (DBSCAN)

O algoritmo de Agrupamento por Densidade (*Density-Based Spatial Clustering of Applications with Noise* - DBSCAN) usa um conceito de densidade para agrupar pontos de acordo com suas vizinhanças (AGGARWAL; REDDY, 2013). Um ponto será considerado denso quando houver vários outros pontos ao seu redor. O algoritmo DBSCAN encontra esses aglomerados de pontos e os coloca em um mesmo *cluster*. Existem dois hiperparâmetros principais: ε (epsilon), que define o tamanho e limites de cada vizinhança, isto é, o quão próximo consideramos ser um ponto vizinho ao outro; e *MinPts* que define o limite da densidade, isto é, quantos pontos são necessários para que seja considerado um grupo. Caso um ponto possua o valor mínimo de *MinPts* em sua vizinhança (epsilon), esses pontos densos serão chamados de pontos centrais. Existem também os pontos de fronteira, que não são densos o suficiente mas pertencem a vizinhança de outro ponto central. Caso um ponto não seja central ou de fronteira, ele é um ponto de ruído (*outlier*).

2.3 Setup experimental

Os algoritmos de agrupamentos foram executados uma única vez, com um *seed* fixo de valor 42. Os algoritmos de agrupamento hierárquico e DBSCAN são determinísticos, ou seja, sempre geram o mesmo valor de agrupamento dado um *seed* fixo. Por outro lado, o KMeans é estocástico, com comportamento dependendo dos pontos iniciais definidos como centróides. Neste artigo não avaliamos o impacto dessas escolhas. Porém, fizemos uma análise inicial para definir a melhor quantidade de *clusters* via regra do cotovelo (MARSLAND, 2009). Esta regra ajuda a encontrar um equilíbrio para formar agrupamentos o mais homogêneos possíveis sendo o mais diferentes possíveis entre eles, assim buscando a quantidade de agrupamentos em que a soma dos quadrados intra-*clusters* seja a menor possível. A análise é descrita na seção de resultados.

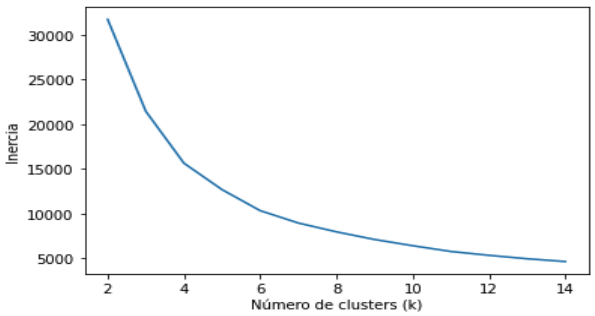
Para avaliar os algoritmos usamos a medida de silhueta (AGGARWAL; REDDY, 2013), já que não existem rótulos no *dataset* para servir de *ground truth*. O coeficiente de silhueta é uma medida de qualidade para toda a estrutura de agrupamento que foi descoberta pelo algoritmo de agrupamento. Os valores podem variar entre -1 e 1, sendo adimensional. Quanto mais próximo de 1, melhor será a qualidade do agrupamento. Os códigos foram gerados na linguagem Python, versão 3.9.4, com uso das bibliotecas Numpy, Pandas, Matplotlib, Scikit-Learn. As configurações da máquina utilizada são Intel core i3-7102E (3M Cache, 2.10 Ghz) com Intel HD graphics 630, placa mãe GIGABYTE GA-H110-D3A, HypeX 4Gb RAM 2400 Mhz, HDD Seagate 1 TB, placa de vídeo GeForce Gt 620, fonte de alimentação Corsair 650 Watts.

3 RESULTADOS

O primeiro experimento realizado foi feito para definir a melhor quantidade de *clusters* para o algoritmo KMeans. Repetimos a aplicação do algoritmo variando a quantidade de *clusters* de 2 até 14. Em cada execução mensuramos o coeficiente de inércia do algoritmo, computado na implementação do scikit-learn. A Figura 1 mostra a curva observada durante esses experimentos. É possível observar que conforme o número de grupos aumenta, o coeficiente de inércia (que indica um equilíbrio entre maior homogeneidade dentro do *cluster* e a maior diferença entre os *clusters*) (MARSLAND, 2009) do agrupamento resultante diminui, mostrando um comportamento de convergência. Para os experimentos,

embora não seja um valor ideal, a escolha de se gerar 4 grupos permite trabalhar com agrupamentos que executam mais rápido e pode apresentar valores razoáveis para as execuções.

Figura 1 –Regra do Cotovelo para KMeans.



Fonte: Autoria própria (2021).

A Tabela 1 mostra os resultados dos coeficientes de silhueta obtidos para cada um dos algoritmos estudados. Os valores obtidos foram semelhantes, porém vale destacar que existem particularidades na manipulação de cada algoritmo. O KMeans depende diretamente da quantidade de grupos que serão gerados, o hiperparâmetro k. Além disso, os centróides são iniciados aleatoriamente de acordo com o *seed* definido para geração de números aleatórios. Com certeza, esse valor de k = 4 influencia o valor da silhueta encontrado.

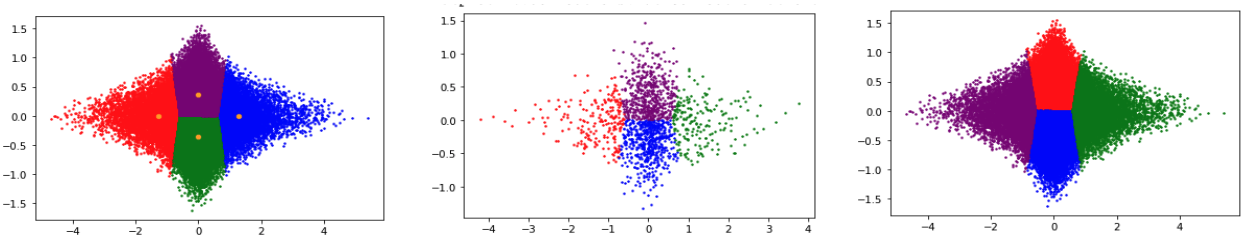
Para o HClust, existe um hiperparâmetro que também determina qual é a quantidade de *clusters* considerados na hierarquia. Aqui, usamos o mesmo valor especificado para o KMeans. O interessante é que o algoritmo de agrupamento hierárquico é muito mais custoso, e para executar apropriadamente, tivemos que realizar uma amostra dos 64 mil exemplos, caso contrário dispende muito tempo. Por fim, o DBSCAN não carece da definição de um número de grupos, apenas dos valores de epsilon e MinPts. Nos experimentos, usamos os valores *default* do scikit-Learn (epsilon = 0.5 e MinPts = 5). O interessante é que o número de *clusters* densos observados foi também igual a 4. Uma visualização dos grupos gerados por cada algoritmo é apresentada na Figura 2.

Tabela 1 – Coeficiente de Silhueta obtido pelos Algoritmos de Agrupamento.

Algoritmo	Silhueta
KMeans	0.361
HClust	0.359
DBSCAN	0.361

Fonte: Autoria própria (2021).

Figura 2 –Resultado dos agrupamentos visualizados em 2 dimensões.



a) KMeans **b) HClust** **c) DBSCAN**

Fonte: Autoria própria (2021).

4 CONCLUSÃO

Neste artigo investigamos o uso de diferentes algoritmos de agrupamento de dados em um problema com a descrição de coordenadas reais de estrelas em um aglomerado. Os experimentos foram realizados com três algoritmos de agrupamento, cada um seguindo um diferente viés de aprendizado, seja ele baseado em distância, na junção/divisão de grupos, ou em densidade. O comparativo realizado permitiu compreender os conceitos de aprendizado não-supervisionado, e deu vislumbres de como realizar uma definição automática de algoritmos para novos problemas. Experimentos futuros almejam atacar esses pontos, por meio de pacotes e ferramentas complementares que determinam o melhor número de *clusters* para um algoritmo. Existem ferramentas na literatura que realizam esse processo para os algoritmos hierárquicos e o DBSCAN, fazendo paralelo à regra do cotovelo empregada no KMeans.

AGRADECIMENTOS

Os autores gostariam de agradecer à Fundação Araucária pelo financiamento do projeto de Iniciação Científica.

REFERÊNCIAS

AGGARWAL, Charu C; REDDY, Chandan K. **Data Clustering**: Algorithms and Applications. Chapman and Hall/CRC, 1 edição, 2013.

LUGER, George F. **Inteligência Artificial** 6° Edição. University of New Mexico, 2013.

MARSLAND, Stephen. **Machine Learning:** An Algorithmic Perspective. Massey University, Palmerston North, New Zealand, 2009.

MITCHELL, Thomas M. **Machine Learning.** McCgraw-Hill, New York, 1997.

RUSSELL, Stuart; NORVIG, Peter. **Inteligência Artificial**. Rio de Janeiro: Elsevier Editora, 2010.

TORGO, L. **Data Mining With R:** Learning with Case Studies, University of Minnesota, 2011.