Trabalho Prático - Algoritmos II
Algoritmos Aproximativos para K-Centros

Rubens da Cunha Castro[1]

[1]Departamento de Ciência da Computação - Universidade Federal de Minas Gerais (UFMG) Belo Horizonte - MG - Brasil

rcastro13@ufmg.br

Abstract. *This report describes the quantitative and qualitative results of two approximation algorithms for the "k-centers"problem. For comparison purposes, the "k-means"algorithm was also evaluated. In all cases, a series of tests was conducted using both real and synthetic data, and evaluation metrics were computed for performance analysis and quality of the results..*

Resumo. *Esse relatório descreve os resultados quantitativos e qualitativos de dois algoritmos aproximativos do problemas de "k-centros". A título de comparação também foi avaliado o algoritmo de "k-means". Em todos esses foram construídos uma bateria de testes utilizando dados reais e fictícios e computadas métricas de avaliação para análise de desempenho e qualidade da resposta.*

1. Introdução

Esse trabalho consistiu no desenvolvimento de três algoritmos com abordagens diferentes para um mesmo problema, o problema dos "k-centros", que recorrentemente aparece em abordagens de clustering, na qual dado um conjunto de pontos ou grafo, que respeitam um conjunto de métricas, incluindo a desigualdade triangular, busca-se verificar se é possível atribuir até k vértices centros que minimizem a distância de um vértice qualquer ao centro.

Dado problema é NP-difícil, o que nos leva a buscar alternativas de resolução para evitar complexidades não polinomiais, e é dentro dessa abordagem que foi aplicado para esse projeto dois algoritmos aproximativos para o problema, sendo um algoritmo "guloso"e o outro baseado no refinamento do intervalo entre os limites inferior e superior do raio. Para efeitos comparativos também foi aplicado o algoritmo do "k-means"para visualização e cálculo da diferença dos raios.

2. Método

2.1. Construção dos Algoritmos

Os algoritmos construídos, como já citado, foram construídos utilizando abordagens gulosa e de refinamento do raio. O primeiro algoritmo possui a simples ideia de escolher o primeiro centro supondo que k = 1 e vai assim por diante até deduzir uma valor máximo de raio. Já no segundo algoritmo, temos a ideia de, dado o intervalo entre 0 e raio máximo, sabe-se que o raio ótimo está contido nesse intervalo, com isso, é possível refinar esse até que se encontre uma margem satisfatória para a resposta. Por fim, tem-se o algoritmo de

K-Means, que aqui mesmo sendo implementado por meio da biblioteca ”sklearn”, a ideia do algoritmo está dentro da lógica de agrupamento dos dados e o raio máximo retornado pode ser utilizado como referência para as soluções que os algoritmos aproximativos vão entregar.

2.2. Construção dos Testes

Após desenvolvimento e validação dos algoritmos foi definido um modelo de geração de amostras para as baterias de testes a serem aplicadas. Primeiramente, foram definidos três grupos de amostras, o primeiro foi obtido por meio de dados reais existentes na plataforma UCI (UCI MACHINE LEARNING REPOSITORY, 2024), escolhendo-se duas colunas de variáveis arbitrárias numéricas e uma coluna de ”label”para 10 bases de dados com mais de 700 instâncias (Tabela 1). Todas as URLs utilizadas foram documentas no repositório desse trabalho.

Os outros dois grupos consistiram em duas modelagens sintéticas. Para o primeiro grupo sintético foram geradas 10 bases de tamanho 800 utilizando a distribuição normal multivariada com o controle do desvio padrão para evitar sobreposição. Para o segundo conjunto foram montadas mais 10 bases de tamanho 500 utilizando datasets preparados pela biblioteca ”sklearn”(SCIKIT-LEARN, 2024) do Python.

Tabela 1. Descrição dos datasets reais utilizados

NOME DATASET	TAMANHO	VARIÁVEIS USADAS	CENTROS
Annealing	798	thick e width	5
Banknote Authentication	1372	variance e skewness	2
Estimation of Obesity Levels Based On Eating Habits and Physical Condition	2111	Age e Height	7
Hepatitis C Virus (HCV) for Egyptian patients	1385	BMI e WBC	4
Rice (Cammeo and Osmancik)	3810	Area e Perimeter	2
Statlog (Landsat Satellite)	6435	Attribute31 e attribute32	6
Waveform Database Generator (Version 1)	5000	Attribute1 e attribute2	3
Wine Quality	4898	chlorides e pH	7
Yeast	1484	mcg e gvh	10
Statlog (German Credit Data)	1000	Attribute2 e attribute13	2

Com todos esses datasets montados e organizados, teve-se à disposição trinta instâncias diferentes para a aplicação dos testes. Para a realização desses, definiu-se as ba-

terias de teste da seguinte forma: trinta execuções para cada instância em cada algoritmo, guloso e refinamento de raio, sendo que para esse segundo seriam testados cinco valores diferentes de porcentagem para o refinamento (1, 2, 4, 8 e 16). Além disso, todos esses testes seriam avaliados primeiro utilizando a distância de Manhattan (p igual a 1) e posteriormente com a Euclidiana (p igual a 2). Para cada teste também foi realizado o cálculo da silhueta, índice que me retorna o quão bem um cada objeto está alocado em um grupo, e do índice de Rand Ajustado, medida de semelhança entre duas diferentes soluções de clustering (alocação verdadeira - labels originais e alocação feita pelo algoritmo).

3. Resultados

Os resultados foram obtidos por meio de uma tabela que foi sendo preenchida em formato textual ao longo da execução dos testes. Cada linha desse arquivo possui o seguinte objeto: [Algoritmo (int), K (int), p (int), instancia (string), tempoExecucao (float), raioSolucao (float), silhueta (float), indiceRand (float), percent (float)]. Após a conversão em ”dataframe”essa tabela foi convertida em um arquivo ”.csv”final que contém o cálculo da média e desvio padrão dos itens de tempo de execução, raio, silhueta e índice de Rand para o agrupamento Algoritmo, instância, ’p’ e ’percent’ (o valor de ’percent’ é útil apenas para o algoritmo de refinamento). No total, o arquivo mostrou que 9900 testes foram executados.

Com base no que foi obtido, os algoritmos serão analisado de maneiras individual, para, por fim, serem comparados ao ”K-Means”.

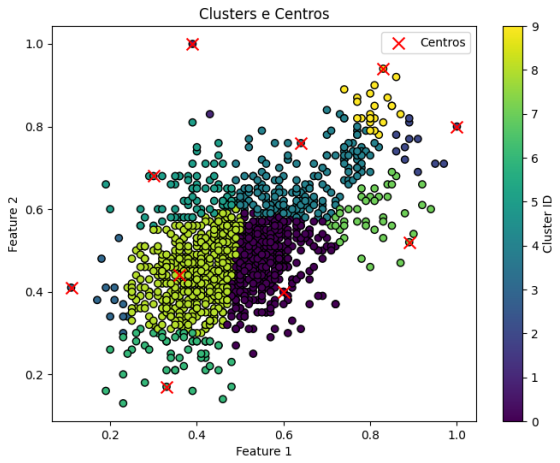

Figura 1. Exemplo de Resposta

3.1. Primeiro Algoritmo (Guloso)

3.1.1. Análise dos Resultados

Esse algoritmo se destacou pela velocidade de execução, pois além de não precisar computar uma matriz de adjâcencia, a escolha arbitrária pelos pontos de centro auxiliam nessa questão performática. Com relação a qualidade da solução, foi percebido que os valores para a distância euclidiana tendem a ser um pouco mais precisos, pois além da solução

com raio menor e menos variada, os valores do índice de Rand e silhuetas são melhores nesses casos (Figura 3 e 4).

Também é perceptível que os indicadores apontam soluções não tão boas para o grupo de testes das instâncias montadas pelo ”sklearn”(SCIKIT-LEARN, 2024). Uma possibilidade de reposta quando a isso é que os pontos desses grupos são mais esparsos e envolvem apenas dois ou três centros, logo, a qualidade da escolha inicial arbitrária pode acabar diminuindo a qualidade da solução.

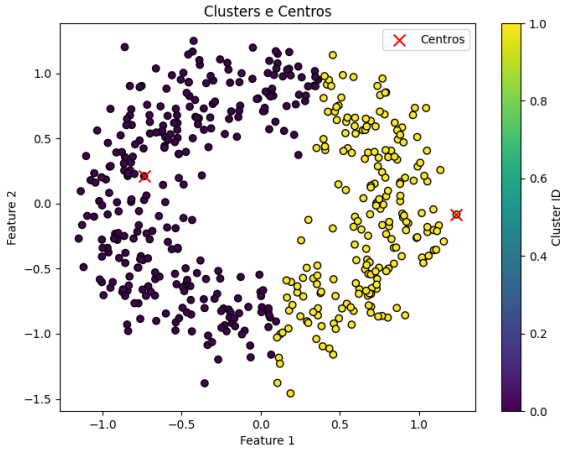

Figura 2. Caso ruim desse Algoritmo

Por fim, as soluções dadas por esse algoritmo variam aproximadamente 10 por cento do valor da média das soluções.

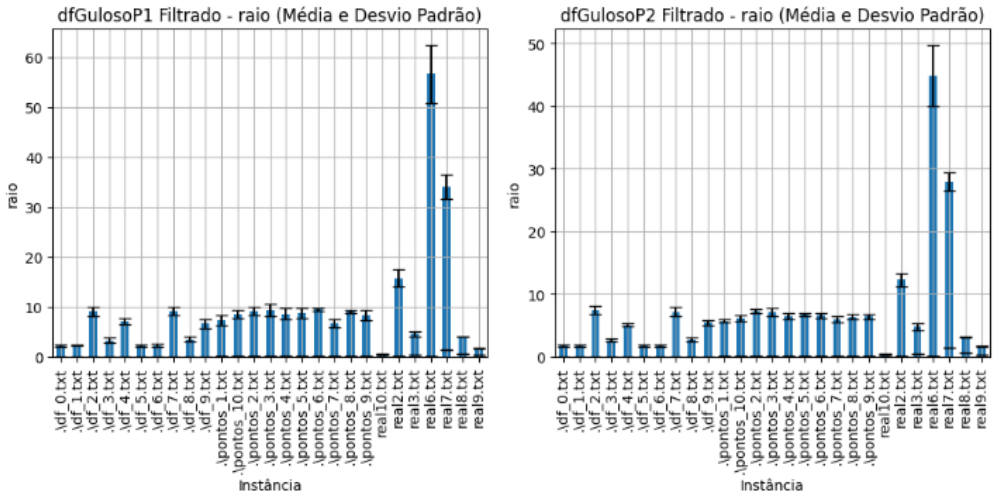

Figura 3. Média e Desvio Padrão das soluções obtidas

3.1.2. Comparação com o K-Means

A partir da comparação das soluções obtidas aqui com o retorno dado pelo K-Means (Figura 5) percebe-se que, de fato, quando o parâmetro p é a distância euclidiana, esse

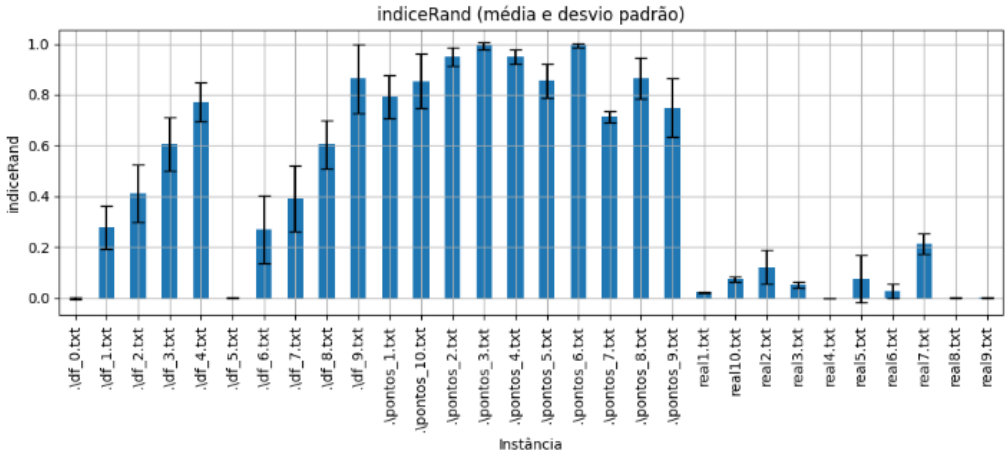

Figura 4. Indice de Rand para p = 1

algoritmo se comporta melhor em termos de qualidade da solução. De maneira geral, a solução dada retonada é satisfatória e traz uma boa aproximação, apesar de ser suscetível a erros maiores dependendo tanto do valor de K quanto da formatação da instância de entrada.

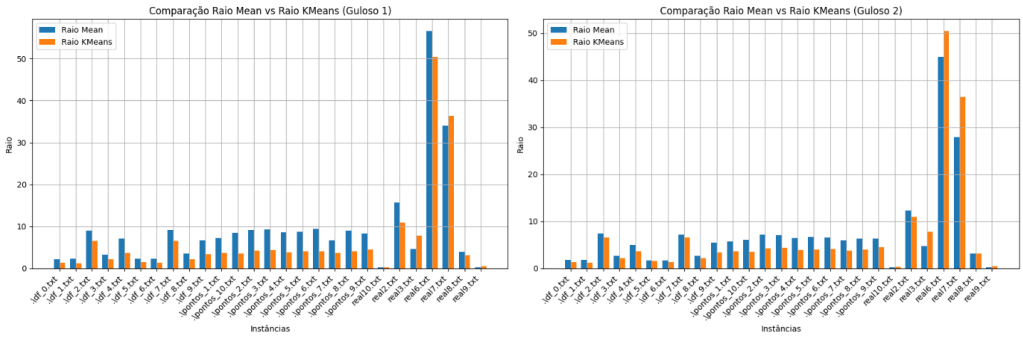

Figura 5. Comparação entre solução do algoritmo e K-Means

3.2. Segundo Algoritmo (Refinamento de Raio)

3.2.1. Análise dos Resultados

Com relação aos testes desse algoritmo temos em termos de desempenho de tempo uma piora em relação ao guloso, muito em função da necessidade de construção da matriz de distâncias entre os pontos. Apesar de na tabela final de resultados esse tempo de construção da matriz ter sido desconsiderado, teve-se em média três minutos para a construção dessa levando em conta as instâncias montadas.

Sobre a solução encontrada (Figura 6), percebe-se que quando maior a porcentagem (variável 'percent') melhor foi a solução retornada pelo algoritmo, o que indica que com o valor ideal desse item, a solução retornada certamente terá um ganho de qualidade (Figura 7). Além disso, apesar das soluções possuírem um alto desvio padrão, é possível perceber que a melhor solução encontrada certamente está bem próxima do valor ótimo.

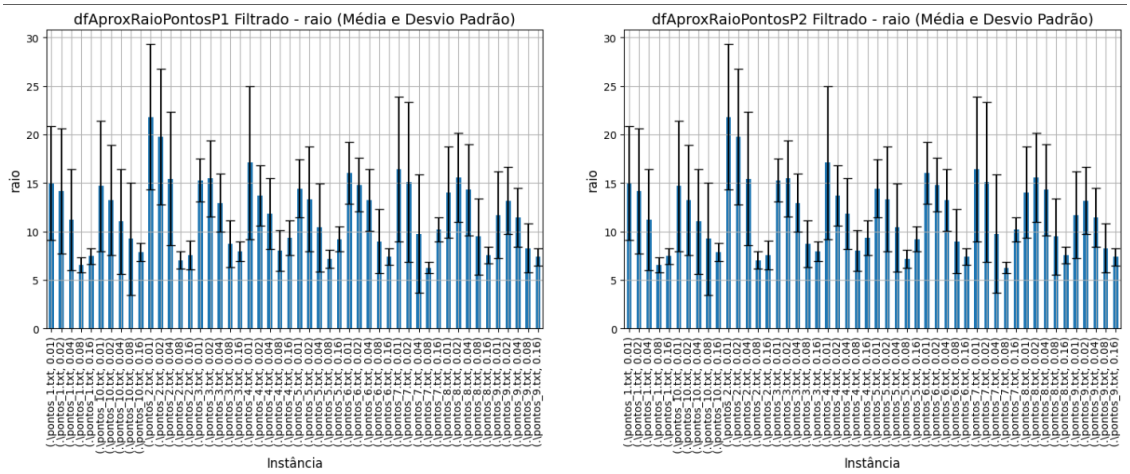

Figura 6. Média e Desvio Padrão das soluções obtidas (dataset sintético)

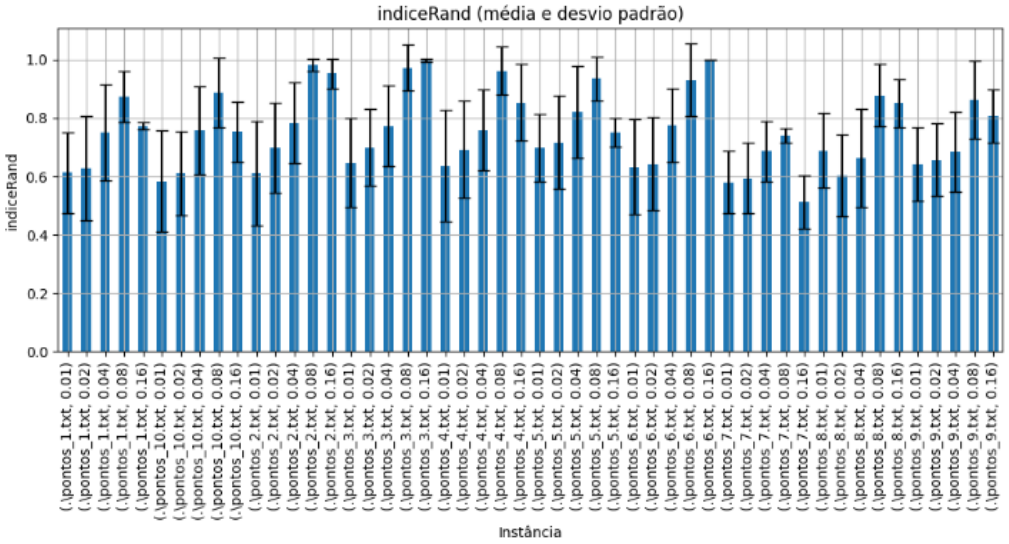

Figura 7. Indice de Rand para p = 1 (dataset sintético)

No mais, uma fraqueza desse algoritmo são instâncias em que a distância máxima é parecida com outros valores de distância entre os pontos, no geral, foram nesses testes que a solução obtida pelo algoritmo mais variou ao longo das trinta execuções (Figura 8).

3.2.2. Comparação com o K-Means

A partir da comparação das soluções obtidas aqui com o retorno dado pelo K-Means (Figura 9) percebe-se que, de fato, quando o parâmetro 'percent' aumenta, mais o retorno do algoritmo se parece com o raio dado pelo K-Means, o que traz o entendimento de que a aproximação é muito boa dados os parâmetros corretos.

3.3. Comparação entre os Algoritmos

De maneira geral, ambos os algoritmos se comportaram de maneira satisfatória e dada a grande quantidade de testes aos quais esses foram submetidos, pode-se ter uma boa ideia

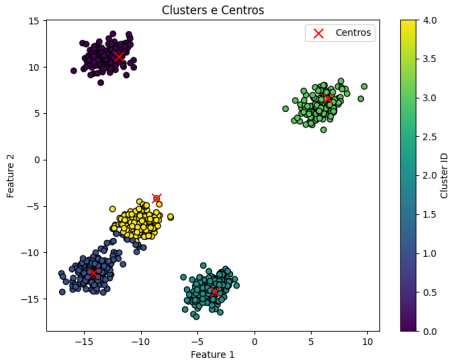

Figura 8. Caso ruim desse Algoritmo

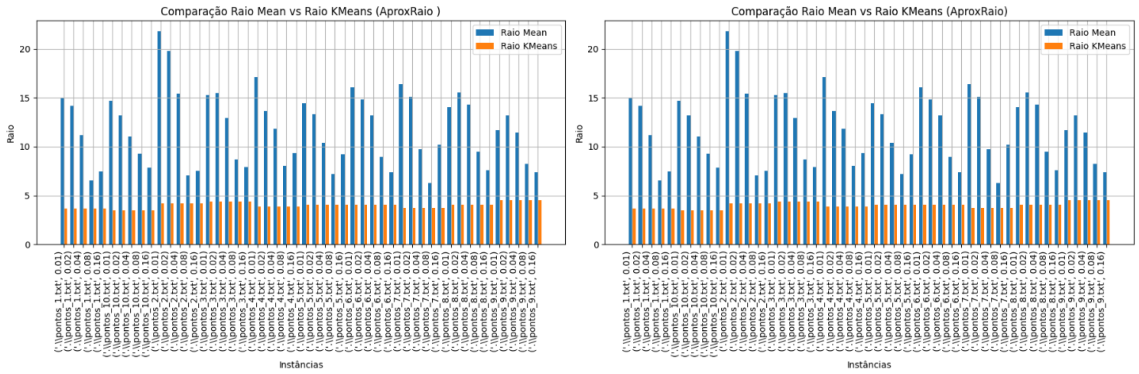

Figura 9. Comparação entre solução do algoritmo e K-Means

dos pontos fortes e fracos de cada método. No que tange ao tempo de execução o primeiro método é firme e consegue manter uma execução rápida mesmo em instâncias maiores. Por outro lado, o método de refinamento é muito custoso para ser executado unitarimente em função da sua matriz de distância, de forma que em instâncias muito grandes pode ser inviável o uso desse. Agora, no que tange a solução obtida, pode-se dar uma nota constante para o método guloso, que mesmo com seu lado fraco em certas instâncias de K pequeno ou com pontos esparsos, ainda consegue dar uma resposta aproximada válida dentro de seus limites. Enquanto isso, o algoritmo de refinamento se mostrou um ótimo algoritmo quando teve parâmetros mais ideais aplicados, levando a soluções que, em um grande conjunto de testes, mesmo apresentando uma grande variância, conseguiu se aproximar mais do retorno dado pelo K-Means como termo comparativo.

4. Conclusão

O estudo de métodos de aproximação, como os que foram aplicados aqui possui grande importância, tanto pela possibilidade de obter soluções válidas para problemas difíceis em tempo polinomial, quanto no estudo da teoria da computação no que tange à caracterização de classes de problemas e limites da computação.

Dessa forma, após a análise e comparação dos algoritmos é possível dizer que o melhor deles é o que melhor se encaixa no seu contexto. De certa maneira, isso traz a flexibilidade de poder ter diversos programas que funcionam em contextos diferentes, seja em termos de poder computacional, seja em termos de análise de instâncias.

Ademais, é preciso dizer que a realização desse trabalho me permitiu entender melhor sobre os algoritmos trabalhados durante as aulas e as particularidades de cada um, o que me fez aprender mais sobre o conteúdo da disciplina e ganhar mais referências e conhecimentos que certamente serão úteis para o desenvolvimento de futuros projetos e trabalhos.

Por fim, de forma a destacar a bibliografia, o trabalho levou a leitura de sites e estruturas de código e possibilitou a análise lógica e computacional dessas estruturas, o que faz referência ao estudo aplicado do curso de "Algoritmos 2" dentro da área de Ciência da Computação no que tange à compreensão e aplicação de algoritmos, identificação e análise de problemas difíceis e a enumerações de soluções aproximadas para esses problemas.

Referências

SCIKIT-LEARN. **Plot Cluster Comparison**. Acesso em: 15 ago. 2024. 2024. Disponível em: ¡https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html#sphx-glr-auto-examples-cluster-plot-cluster-comparison-py¿.

UCI MACHINE LEARNING REPOSITORY. **UCI Machine Learning Repository**. Acesso em: 15 ago. 2024. 2024. Disponível em: ¡https://archive.ics.uci.edu¿.