Universidade Federal do Rio Grande do Norte

Departamento de Engenharia Elétrica

Introdução à Agrupamento de Dados (*Data Clustering*) usando Python

Compilação de materiais e códigos por

José Alfredo F. Costa e Esdras Daniel Souza A. Silva

Natal, setembro de 2023

SUMÁRIO

PREFÁCIO

A clusterização é uma área fundamental da análise de dados, permitindo identificar padrões e estruturas ocultas em conjuntos de dados complexos. Este trabalho descreve alguns dos principais conceitos, algoritmos e aplicações da clusterização, tanto métodos tradicionais como recentes baseados em deep learning.

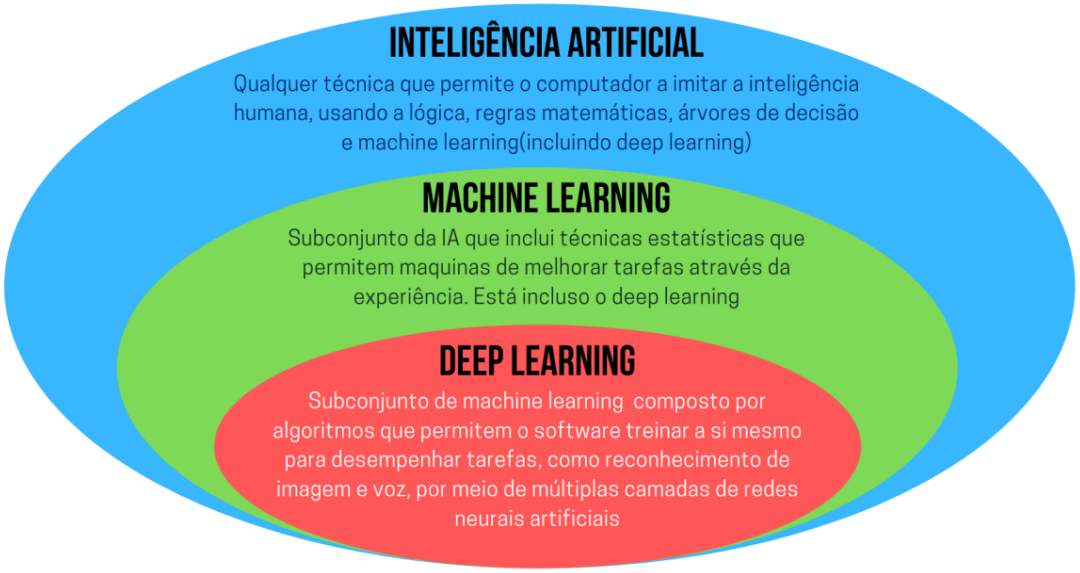
Apresenta-se uma visão geral didática (introdutória) de alguns algoritmos de clusterização, incluindo K-means, Affinity Propagation, Mean Shift, DBSCAN e Spectral Clustering. Também foram discutidos métodos modernos que empregam redes neurais profundas para aprimorar o processo de clusterização.

Exemplos de aplicações são descritos, os pseudocódigos e os códigos em Python, que estarão disponíveis para testes.

APRENDIZADO DE MÁQUINA

Aprendizado de máquina (ou Machine Learning em inglês) é uma subárea da inteligência artificial que se concentra no desenvolvimento de algoritmos e modelos que permitem que computadores aprendam e melhorem seu desempenho em uma tarefa específica por meio de experiência e dados, em vez de serem explicitamente programados para essa tarefa [1] O aprendizado de máquina está revolucionando diversas áreas, como visão computacional [2], processamento de linguagem natural [3], robótica [4] e reconhecimento de padrões [5]. Algumas aplicações incluem algoritmos de recomendação, chatbots, diagnóstico médico assistido e carros autônomos.

O objetivo principal do aprendizado de máquina é permitir que os computadores aprendam com os dados disponíveis e identifiquem padrões, relações ou características ocultas nesses dados, a fim de tomar decisões, fazer previsões ou realizar tarefas específicas sem intervenção humana constante. Dentro do aprendizado de máquina podemos destacar duas áreas: Aprendizado Supervisionado e o Aprendizado Não Supervisionado [6]:



Aprendizado Supervisionado

O aprendizado de máquina supervisionado é um tipo de aprendizado em que um algoritmo é treinado para aprender a mapear um conjunto de entradas para suas respectivas saídas, com base em exemplos rotulados fornecidos durante o treinamento. Em outras palavras, o algoritmo recebe dados de entrada e sua saída correta associada, e seu objetivo é aprender a relação entre as entradas e as saídas para que, quando encontrar novos dados de entrada, possa prever a saída correta [7].

Para melhor compreensão, vamos ilustrar esse conceito com um exemplo:

Suponha que temos um conjunto de dados que representa informações sobre diferentes frutas. Cada entrada contém características da fruta, como cor, tamanho e textura, e a saída associada é o nome da fruta.

Conjunto de dados

Características	Fruta
Vermelho, Pequeno, Liso	Maçã
Amarelo, Pequeno, Liso	Banana
Amarelo, Grande, Rugoso	Abacaxi
Laranja, Pequeno, Liso	Laranja
Vermelho, Grande, Liso	Maçã
Verde, Pequeno, Liso	Kiwi

Passos do aprendizado de máquina supervisionado:

1. **Preparação dos dados de treinamento:** Os dados são divididos em duas partes: as características de entrada (X) e as saídas esperadas (y). Para o exemplo acima, as características seriam X = {Vermelho, Amarelo, Laranja ou Verde; Pequeno ou Grande; Liso ou Rugoso} e as saídas esperadas seriam y = {Maçã, Banana, Abacaxi, Laranja ou Kiwi}.
2. **Treinamento do algoritmo:** O algoritmo de aprendizado de máquina é alimentado com os dados de treinamento (X e y). Ele examina as características e suas saídas associadas, procurando padrões e relações que possam descrever como as características determinam a fruta correta. Durante esse processo, o algoritmo ajusta seus parâmetros internos para melhor representar essas relações.
3. **Teste do modelo:** Após o treinamento, o modelo resultante é testado em um conjunto de dados de teste separado, que não foi visto durante o treinamento. Isso é importante para verificar se o modelo é capaz de generalizar bem e fazer previsões precisas em novos dados.
4. **Predição em novos dados:** Após passar no teste, o modelo está pronto para ser usado em dados não rotulados. Por exemplo, se apresentarmos ao modelo um conjunto de características de uma fruta desconhecida, como "Amarelo, Pequeno, Rugoso", ele deverá prever que se trata de um "Abacaxi" com base no aprendizado anterior.

O aprendizado de máquina supervisionado é muito útil quando temos muitos dados rotulados e desejamos automatizar tarefas de classificação ou previsão em novos dados, tornando-o um componente importante em diversas aplicações práticas.

Aprendizado Não Supervisionado

O aprendizado de máquina não supervisionado é uma abordagem em que um algoritmo é treinado para encontrar padrões, estruturas ou representações ocultas nos dados, sem utilizar rótulos ou saídas conhecidas. Diferentemente do aprendizado supervisionado, aqui o algoritmo não recebe exemplos de entrada e saída correspondentes para aprender, mas, em vez disso, explora a estrutura intrínseca dos dados para identificar agrupamentos, tendências ou anomalias [8].

Vamos explorar esse conceito com um exemplo:

Suponha que temos um conjunto de dados que contém informações sobre clientes de um supermercado. Cada entrada representa um cliente com base em suas compras, mas não temos rótulos ou categorias predefinidas.

Conjunto de dados

Clientes	Valor das Compras (R$)	Visitas mensais
Cliente 1	50	4
Cliente 2	70	5
Cliente 3	30	2
Cliente 4	100	8
Cliente 5	120	7
Cliente 6	40	3

Passos do aprendizado de máquina não supervisionado:

1. **Preparação dos dados:** Organizamos os dados de entrada (neste caso, as colunas Compras e Visitas Mensais) em uma matriz X, para que o algoritmo possa analisá-los.
2. **Treinamento do algoritmo não supervisionado:** Utilizando um algoritmo de aprendizado não supervisionado, como um algoritmo de agrupamento (clustering) , por exemplo, o modelo é alimentado com o conjunto de dados X. O objetivo do algoritmo é encontrar padrões nos dados, agrupando os clientes em clusters (grupos) com base em sua similaridade. Ele explora a proximidade entre os clientes com base nas características fornecidas (compras e visitas mensais) para formar os clusters.
3. **Identificação de clusters:** Após o treinamento, o algoritmo segmenta os clientes em diferentes clusters com base nas características de compras e visitas mensais. Por exemplo, podemos obter três clusters: Cluster 1 com clientes que fazem compras menores e visitam o supermercado com pouca frequência; Cluster 2 com clientes que

fazem compras moderadas e visitam o supermercado regularmente; e Cluster 3 com clientes que fazem compras grandes e visitam o supermercado com alta frequência.

4. **Análise dos resultados:** Após a formação dos clusters, podemos explorar os padrões identificados e entender as características de cada grupo de clientes. Isso pode nos ajudar a segmentar a base de clientes, direcionar estratégias de marketing específicas para cada grupo ou entender o comportamento de compra dos clientes.

O aprendizado de máquina não supervisionado é especialmente útil quando não temos rótulos para os dados ou quando queremos descobrir informações ocultas que não são óbvias à primeira vista. Ele é amplamente aplicado em tarefas como segmentação de clientes, detecção de anomalias, redução de dimensionalidade, entre outras áreas onde a estrutura dos dados precisa ser explorada sem a necessidade de saídas rotuladas. A área de clusterização, foco deste trabalho, é uma técnica central do aprendizado não supervisionado, permitindo identificar e analisar padrões e estruturas ocultas em conjuntos de dados complexos [9]. Os algoritmos de clusterização serão explorados em detalhes nas próximas seções.

ALGORITMOS DE CLUSTERIZAÇÃO

Existe uma ampla variedade de algoritmos para realizar a tarefa de agrupamento de dados (clusterização). Cada técnica tem suas próprias suposições, vantagens e limitações. A seguir, apresentaremos uma visão geral de alguns dos principais algoritmos de clusterização [10]:

K-means

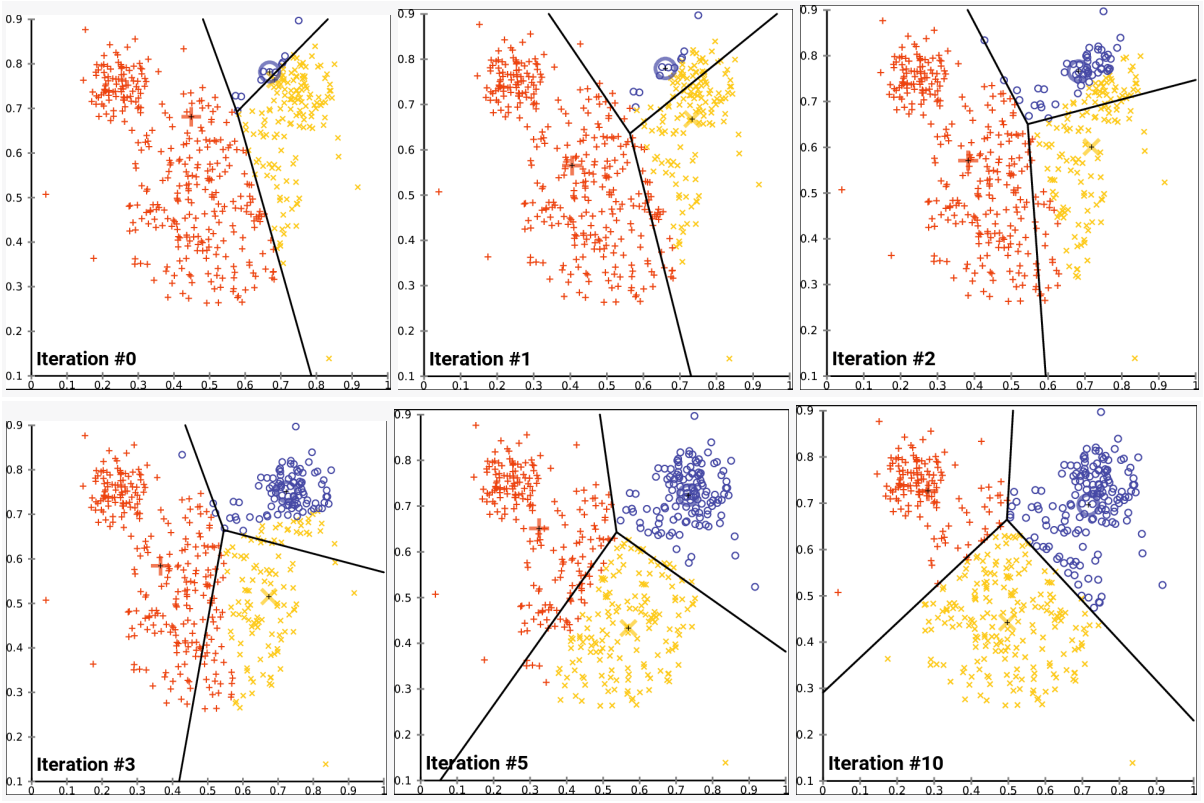
O algoritmo de clusterização K-means [11] é um dos métodos mais populares e eficientes para agrupar dados em clusters. Ele é uma técnica que procura particionar um conjunto de dados em K grupos distintos, onde cada grupo é representado por um centróide, que é a média dos pontos no cluster. O objetivo do algoritmo é minimizar a variância interna dos clusters, ou seja, tentar agrupar os pontos de modo que eles sejam semelhantes dentro de cada grupo e diferentes entre os grupos.

Passo a passo de como o algoritmo K-means funciona:

1. **Inicialização:** O algoritmo começa selecionando aleatoriamente K centróides, onde K é o número de clusters que desejamos obter. Esses centróides podem ser escolhidos a partir de amostras aleatórias dos dados ou de qualquer outra estratégia de inicialização.
2. **Atribuição de pontos ao cluster mais próximo:** Para cada ponto do conjunto de dados, o algoritmo calcula a distância entre o ponto e cada um dos centróides. O ponto é atribuído ao cluster cujo centróide está mais próximo, com base na distância euclidiana (ou outra medida de distância).
3. **Atualização dos centróides:** Uma vez que todos os pontos foram atribuídos aos clusters, o algoritmo calcula os novos centróides para cada grupo, calculando a média das coordenadas de todos os pontos no cluster.
4. **Reatribuição de pontos:** O algoritmo repete o processo de atribuição dos pontos ao cluster mais próximo, considerando os novos centróides calculados no passo anterior. Isso pode levar a uma mudança na composição dos clusters.
5. **Convergência:** Os passos 3 e 4 são repetidos em um loop até que os centróides praticamente não mudem entre iterações ou até que uma condição de parada pré-definida seja satisfeita (por exemplo, um número máximo de iterações).

O algoritmo de clusterização K-Means é popular por sua simplicidade e eficiência em dividir um conjunto de dados em grupos coesos, ser computacionalmente eficiente [12] e seu resultado é fácil de interpretar. Porém, a eficácia do K-Means depende da escolha do número correto de clusters, algo que pode ser subjetivo, a posição dos centróides iniciais podem levar a diferentes soluções finais, o que torna a inicialização crucial. Também assume que os clusters são esféricos, o que pode não se adequar bem a dados com formas mais complexas, além de que outliers podem afetar significativamente a formação dos clusters.

Figura 1 - Iterações do algoritmo K-means



Affinity Propagation

O algoritmo de clusterização Affinity Propagation [13] é uma técnica de aprendizado não supervisionado que não requer a especificação prévia do número de clusters K. Ele é capaz de encontrar automaticamente o número de clusters e determinar quais pontos devem ser os exemplares (representantes) de cada cluster.

O Affinity Propagation é especialmente útil quando os dados têm uma estrutura complexa e não se enquadram facilmente em clusters convexos ou bem definidos. Ele é amplamente utilizado em várias áreas, como análise de redes sociais, reconhecimento de padrões, análise de imagens e bioinformática.

Descrição passo a passo de como o algoritmo Affinity Propagation funciona:

1. **Similaridade entre pontos:** O primeiro passo do algoritmo é calcular uma matriz de similaridade entre todos os pontos do conjunto de dados. Essa matriz de similaridade representa o quanto cada ponto é similar a todos os outros pontos do conjunto. A similaridade pode ser medida usando uma métrica como a distância euclidiana, por exemplo.
2. **Escolha dos exemplares iniciais (Responsabilidade e Disponibilidade):** Cada ponto inicialmente considera todos os outros pontos como seus possíveis exemplares e atualiza duas matrizes, a matriz de Responsabilidade (R) e a matriz de Disponibilidade (A). A matriz de Responsabilidade (R) contém informações sobre o

quanto cada ponto acredita que outros pontos devem ser seus exemplares, com base na similaridade. A matriz de Disponibilidade (A) contém informações sobre o quanto cada ponto está disponível para ser um exemplar, com base na similaridade dos pontos que o consideram um exemplar.

3. **Atualização das matrizes R e A:** O algoritmo atualiza iterativamente as matrizes R e A até que a convergência seja alcançada. Durante cada iteração, a matriz de Responsabilidade é atualizada para refletir a similaridade entre pontos e seus exemplares. A matriz de Disponibilidade é atualizada para refletir a disponibilidade de pontos para serem exemplares, considerando a responsabilidade dos outros pontos.
4. **Determinação dos clusters:** Uma vez que as matrizes R e A convergirem, os exemplares finais são determinados. Os pontos que têm alta disponibilidade são escolhidos como exemplares e são os representantes de seus clusters. Os pontos que têm alta responsabilidade em relação a esses exemplares também são atribuídos a esses clusters.
5. **Resultado:** Os pontos são agrupados nos clusters definidos pelos exemplares. Cada cluster representa um grupo de pontos que compartilham uma forte afinidade entre si.

Quando você não tem uma ideia clara sobre quantos clusters existem nos seus dados, este algoritmo pode ser vantajoso, já que ele não requer uma predefinição do número de clusters. O Affinity Propagation, além de poder capturar relações complexas entre os dados, especialmente quando os clusters têm formas não-lineares ou diferentes tamanhos, também é mais robusto a outliers em comparação com alguns outros algoritmos de clusterização, devido à sua estrutura de cálculo de afinidade. Porém, o Affinity Propagation pode ser computacionalmente caro, especialmente para conjuntos de dados grandes. A matriz de similaridade e a matriz de responsabilidades podem se tornar grandes, resultando em um consumo considerável de memória e tempo de processamento. Além disso, uma escolha inadequada dos parâmetros do algoritmo pode levar a resultados subótimos ou até mesmo à convergência falha.

Ref:
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.490.7628&rep=rep1&type=pdf

Mean Shift

O algoritmo de clusterização Mean Shift [14] é uma técnica de aprendizado não supervisionado que agrupa os dados em clusters com base na densidade dos pontos. Ao contrário do K-means, o Mean Shift não requer a especificação prévia do número de clusters K. Ele pode identificar automaticamente o número de clusters com base nas características dos dados.

A ideia principal por trás do algoritmo Mean Shift é deslocar iterativamente os pontos em direção a regiões mais densas do espaço de características até que eles se estabilizem em regiões de alta densidade, que correspondem aos centros dos clusters.

Descrição passo a passo de como o algoritmo Mean Shift funciona:

1. **Escolha dos centros iniciais:** O algoritmo começa selecionando aleatoriamente ou através de alguma estratégia de inicialização os pontos iniciais que atuarão como centros dos clusters.
2. **Cálculo das direções e distâncias:** Para cada ponto central escolhido no passo anterior, o algoritmo calcula as direções e as distâncias em relação a outros pontos do conjunto de dados. Isso é feito com base em uma função de kernel que avalia a proximidade dos pontos. A função mais comumente utilizada é a "flat" que atribui o mesmo peso a todos os pontos dentro de uma determinada distância do centro. Essa distância é definida como o raio do kernel. Todos os pontos dentro do raio do kernel têm a mesma importância na estimativa da densidade ao redor do centro.
3. **Deslocamento para regiões de alta densidade:** Os pontos são deslocados iterativamente em direção às regiões de maior densidade (onde a concentração de pontos é maior). Esse deslocamento é realizado calculando-se o vetor médio das direções e distâncias obtidas no passo anterior. Esse vetor representa o novo deslocamento para cada ponto central.
4. **Convergência:** O algoritmo repete o passo 2 e 3 até que os pontos se estabilizem em regiões de alta densidade, ou seja, até que os pontos centralizados não se movam mais significativamente.
5. **Atribuição aos clusters:** Uma vez que os pontos centralizados se estabilizam, o algoritmo atribui cada ponto do conjunto de dados ao cluster cujo ponto central está mais próximo.

Sendo assim, o algoritmo Mean Shift é eficaz na identificação de clusters com formas não-lineares e complexas, o que pode ser difícil para métodos como o K-Means. Porém o desempenho do Mean Shift pode depender da escolha adequada do bandwidth, que pode ser um parâmetro sensível. Também, em alguns casos, o Mean Shift pode convergir para centros locais, resultando em uma solução subótima.

Ref.:
https://towardsdatascience.com/understanding-mean-shift-clustering-and-implementation-with-python-6d5809a2ac40

Spectral Clustering

O algoritmo de clusterização Spectral Clustering [15] é uma técnica avançada de aprendizado de máquina não supervisionado que utiliza conceitos da teoria dos grafos e da análise espectral para agrupar os dados em clusters. Ele é especialmente útil quando os dados não podem ser facilmente separados por fronteiras lineares, pois permite capturar estruturas complexas e não lineares nos dados.

Descrição passo a passo de como o algoritmo de clusterização Spectral Clustering funciona:

1. **Construção do Grafo de Similaridade:** Primeiro, se cria um grafo onde os nós representam os dados que se deseja agrupar. A similaridade entre os pontos é usada para conectar esses nós com arestas. Quanto mais semelhantes forem dois pontos, mais forte é a aresta que os conecta.
2. **Matriz Laplaciana:** Com base no grafo de similaridade, é criada a matriz Laplaciana. Essa matriz resume a estrutura do grafo, representando a conectividade entre nós.
3. **Decomposição Espectral:** Agora, se calcula os autovetores e autovalores da matriz Laplaciana. A decomposição espectral envolve dividir a matriz em suas partes constituintes.
4. **Redução de Dimensionalidade:** Depois de obter os autovetores, o próximo passo é reduzir a dimensionalidade dos dados. Escolhe-se os autovetores associados aos menores autovalores, que capturam as variações de baixa frequência nos dados. Esses autovetores transformam os dados em um espaço de dimensionalidade reduzida.
5. **Clusterização:** Por fim, a clusterização é realizada nos dados reduzidos. Algoritmos tradicionais de clusterização, como K-means, podem ser aplicados nessa nova representação dos dados para agrupá-los em clusters.

O Spectral Clustering é poderoso porque pode lidar com dados não lineares e capturar estruturas complexas em conjuntos de dados. No entanto, ele também requer o ajuste de parâmetros importantes, como o número de clusters e a escolha da função de similaridade ou distância, que podem afetar significativamente os resultados obtidos.

Agglomerative clustering

O algoritmo de clusterização aglomerativa, também conhecido como Agglomerative Clustering, é uma técnica de aprendizado de máquina não supervisionado que agrupa os dados em uma hierarquia de clusters "bottom-up" (começa com cada ponto sendo seu próprio cluster e, em seguida, mescla iterativamente os clusters). O Agglomerative Clustering é uma abordagem genérica que permite escolher diferentes critérios de mesclagem para formar os clusters. Existem quatro critérios de mesclagem comuns utilizados nessa abordagem: ligação única (single linkage), ligação completa (complete linkage), ligação média (average linkage) e Ward.

- Na **ligação única**, a distância entre dois clusters é definida como a menor distância entre qualquer par de pontos pertencentes a cada cluster. Isso significa que os dois clusters são mesclados com base nos dois pontos mais próximos entre eles.
- Na **ligação completa**, a distância entre dois clusters é definida como a maior distância entre qualquer par de pontos pertencentes a cada cluster. Isso significa que os dois clusters são mesclados com base nos dois pontos mais distantes entre eles.
- Na **ligação média**, a distância entre dois clusters é definida como a média de todas as distâncias entre os pontos pertencentes a cada cluster. Isso significa que os dois

clusters são mesclados com base na média das distâncias entre todos os pontos de ambos os clusters.

- A l**igação Ward** foca na minimização da variância dentro dos clusters. Ele calcula a soma das diferenças quadráticas entre cada ponto e a média do cluster a que pertence, para todos os clusters que seriam formados pela mesclagem. A mesclagem que resulta na menor soma de diferenças quadráticas é escolhida. Isso tende a formar clusters compactos e homogêneos.

  Passo a passo de como o algoritmo de clusterização aglomerativa funciona:

1. **Inicialização:** Inicialmente, cada ponto de dado é considerado como um cluster individual.
2. **Cálculo da matriz de distância:** Em seguida, é calculada uma matriz de distância que contém as distâncias entre todos os pares de pontos de dados no conjunto de dados. A distância pode ser medida usando uma métrica como a distância euclidiana ou uma função de similaridade.
3. **Mesclagem dos Clusters:** O algoritmo começa mesclando os dois clusters mais próximos, ou seja, aqueles que têm a menor distância entre eles. A distância entre dois clusters é definida como a soma dos quadrados das diferenças entre todos os pontos dos clusters e o centro do novo cluster mesclado.
4. **Atualização da Matriz de Distância:** Após mesclar dois clusters, a matriz de distância é atualizada para refletir a nova distância entre os clusters mesclados e os clusters restantes. O critério de ligação de Ward considera a variação dos pontos dentro dos clusters e entre os clusters mesclados para calcular a distância.
5. **Repetição:** Os passos 3 e 4 são repetidos iterativamente até que todos os pontos estejam em um único cluster. Isso forma uma hierarquia de clusters, representada por um dendrograma.
6. **Dendrograma e Corte:** O dendrograma representa a hierarquia de clusters em forma de árvore. Para determinar o número de clusters desejado, pode-se fazer um "corte" no dendrograma em um determinado nível, que corresponderá ao número de clusters desejado.

Cada critério de linkage tem suas próprias características e é mais adequado para diferentes tipos de dados e estruturas de cluster. Não há um critério de ligação universalmente superior, e a escolha depende do contexto da análise e dos objetivos específicos. Muitas vezes, a experimentação com diferentes critérios é necessária para determinar qual critério se ajusta melhor aos seus dados e ao resultado desejado.

DBSCAN

O algoritmo de clusterização DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [16] é um método de aprendizado de máquina não supervisionado que agrupa os dados com base na densidade dos pontos. Ao contrário de outros algoritmos de clusterização, como o K-means, o DBSCAN não requer a especificação prévia do número de

clusters K. Ele é especialmente útil para identificar clusters em conjuntos de dados com formas irregulares e que contêm ruídos.

Passo a passo de como o algoritmo de clusterização DBSCAN funciona:

1. **Determinação de pontos centrais e pontos de borda:** O algoritmo começa escolhendo um ponto qualquer do conjunto de dados que ainda não foi visitado e verifica os seus pontos vizinhos (dentro de uma determinada distância chamada de raio de vizinhança). Se na sua vizinhança contiver um número mínimo de pontos (chamado de "*minPts*"), o ponto inicial é considerado um "ponto central" caso contrário ele é considerado um “ponto de borda”.
2. **Expansão do cluster:** A partir de um ponto central o algoritmo expande o cluster incluindo todos os pontos alcançáveis que pertencem à sua vizinhança. Os pontos centrais que foram inseridos no agrupamento também vão expandir o cluster incluindo seus pontos vizinhos, já os pontos de borda que foram inseridos não vão inserir seus vizinhos.
3. **Expansão para outros pontos não visitados:** O algoritmo continua repetir o passo 2 para outros pontos não visitados até que todos os pontos tenham sido visitados.
4. **Identificação de ruídos:** Caso haja pontos que não foram inseridos em nenhum clusters eles serão identificados como outliers..

O DBSCAN é capaz de identificar clusters com formas irregulares e é robusto em relação a ruídos, pois os pontos isolados são considerados ruídos e não são atribuídos a nenhum cluster. Ele também não requer a definição prévia do número de clusters, o que é uma grande vantagem em relação a outros algoritmos. No entanto, a escolha adequada dos parâmetros, como o raio de vizinhança e o número mínimo de pontos (*minPts*), é importante para obter resultados ótimos.

OPTICS

O algoritmo de clusterização OPTICS (Ordering Points To Identify the Clustering Structure) é um método de aprendizado de máquina não supervisionado que realiza a clusterização de dados baseado na densidade dos pontos. O OPTICS é uma extensão do DBSCAN e pode identificar clusters de diferentes tamanhos e densidades.

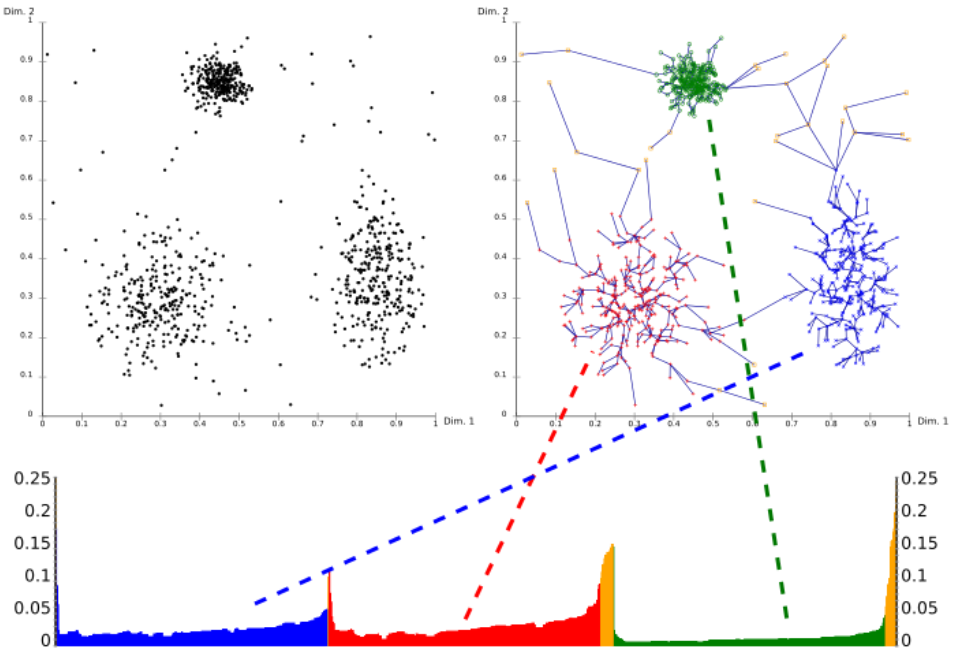
Descrição passo a passo de como o algoritmo de clusterização OPTICS funciona:

1. **Escolha de um Ponto Inicial:** O algoritmo inicia selecionando um ponto qualquer do conjunto de dados que ainda não tenha sido visitado.
2. **Exploração da Vizinhança e Construção do Grafo de Alcance:** O algoritmo verifica os pontos vizinhos desse ponto inicial dentro de um determinado raio de vizinhança, similar ao DBSCAN. Ao contrário do DBSCAN, em vez de atribuir os pontos imediatamente a um cluster, o OPTICS cria uma estrutura chamada de grafo de alcance (reachability plot).

3. **Construção do Reachability Plot:** O grafo de alcance é uma representação da densidade dos pontos em relação à distância. Ele armazena informações sobre a densidade dos pontos dentro de uma determinada distância (alcance) a partir do ponto inicial. Essa estrutura é usada para identificar os clusters de diferentes densidades.
4. **Ordenação dos Pontos:** O OPTICS utiliza o grafo de alcance para ordenar os pontos do conjunto de dados. A ordem é baseada em dois fatores: a densidade local (número de pontos dentro do alcance) e a acessibilidade (alcance) aos pontos com densidades maiores. Essa ordenação ajuda a identificar a estrutura de clusterização nos dados.
5. **Extrair Clusters:** A partir da ordenação dos pontos, o algoritmo extrai os clusters identificando os pontos que possuem alcances significativos (representativos de agrupamentos densos). Através dessa informação, o OPTICS também pode identificar os pontos de ruído e outliers.

O algoritmo de clusterização OPTICS é útil em cenários onde os dados possuem clusters de diferentes tamanhos e densidades, pois ele pode adaptar-se a diferentes estruturas de clusterização. Sua capacidade de criar o grafo de alcance e ordenar os pontos baseados em densidade e acessibilidade permite uma análise mais completa da estrutura dos dados. Assim como o DBSCAN, o OPTICS também é eficiente em lidar com pontos de ruído e outliers, contribuindo para a robustez da clusterização.

Figura 2 - Exemplo de clusters e Reachability plot



GMMs (Gaussian Mixture Models)

Os Modelos de Mistura Gaussiana (Gaussian Mixture Models, GMMs) são uma poderosa técnica de aprendizado de máquina usada principalmente para modelagem de

densidade de probabilidade e tarefas de clusterização. Eles são particularmente eficazes quando os dados subjacentes têm uma estrutura complexa e não podem ser adequadamente modelados por um único modelo de distribuição.

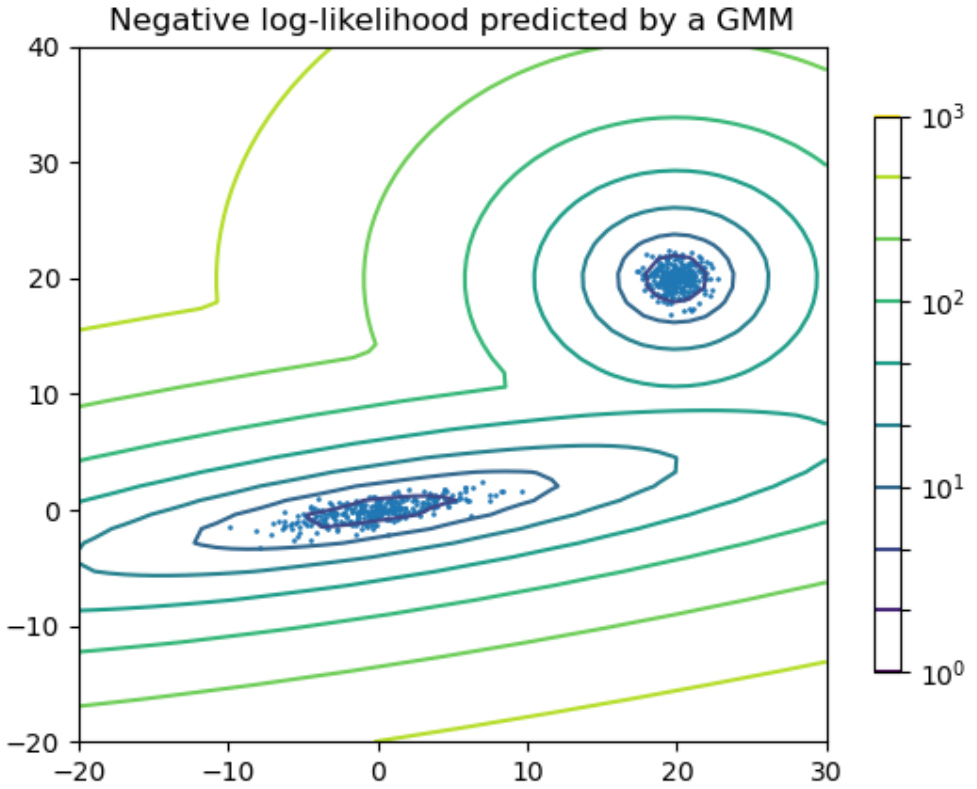
A ideia fundamental por trás dos GMMs é que um conjunto de dados pode ser representado como uma combinação ponderada de várias distribuições gaussianas (também conhecidas como normais). Cada uma dessas distribuições gaussianas representa um cluster ou componente latente dos dados. Portanto, em vez de assumir que os dados pertencem a um único cluster, os GMMs permitem que os dados sejam uma mistura de várias distribuições gaussianas.

Descrição passo a passo de como o algoritmo Gaussian Mixture funciona:

1. **Inicialização:** Primeiro, você precisa iniciar os parâmetros do modelo. Isso inclui o número de componentes gaussianos (clusters) a serem usados e as estimativas iniciais dos parâmetros para cada componente, como médias, covariâncias e pesos.
2. **Expectation-Maximization (EM):** A etapa mais crucial na modelagem de GMMs é a aplicação do algoritmo Expectation-Maximization. Este é um algoritmo iterativo que é usado para ajustar os parâmetros do modelo com base nos dados de treinamento.
    - **Expectation (E-step):** Nesta etapa, você calcula as probabilidades de pertencimento de cada ponto de dados a cada componente gaussiano. Isso é feito usando a função de densidade de probabilidade gaussiana e os parâmetros atuais do modelo.
    - **Maximization (M-step):** Na etapa de maximização, você atualiza os parâmetros do modelo, ou seja, as médias, covariâncias e pesos, com base nas probabilidades calculadas no passo anterior.
3. **Convergência:** O processo de E-M é repetido iterativamente até que o modelo alcance a convergência, ou seja, até que os parâmetros não mudem significativamente entre iterações subsequentes.
4. **Atribuição de Cluster:** Depois de treinar o modelo GMM, você pode usá-lo para atribuir pontos de dados a clusters. Isso é feito calculando a probabilidade de pertencimento de cada ponto a cada componente gaussiano e atribuindo o ponto ao cluster com a probabilidade mais alta.

Os GMMs têm várias vantagens, incluindo a capacidade de modelar distribuições de dados complexas e sobrepostas. No entanto, eles também têm algumas limitações, como a sensibilidade à inicialização e a necessidade de especificar o número de componentes gaussianos. Portanto, a seleção adequada do número de clusters é uma etapa crítica no uso de GMMs para clusterização.

Figura 3 - Exemplo de GMMs



BIRCH

O BIRCH, que significa "Balanced Iterative Reducing and Clustering using Hierarchies" (Redução e Clusterização Balanceadas e Iterativas Usando Hierarquias), é um método de clusterização popular que foi projetado para lidar eficientemente com grandes conjuntos de dados. Ele é especialmente adequado para dados que não cabem na memória principal do computador, tornando-o útil em cenários de big data e mineração de dados.

Descrição geral do método de clusterização BIRCH:

1. **Agrupamento Hierárquico:** O BIRCH utiliza uma abordagem hierárquica para a clusterização. Ele começa com uma estrutura hierárquica vazia, que é construída à medida que os dados são processados.
2. **Repartição do Espaço de Características:** O espaço de características (onde os dados são representados) é dividido em subespaços, e os dados são agrupados em subgrupos chamados "subclusters" em cada subespaço. Isso é feito para reduzir a complexidade computacional e a quantidade de dados a serem mantidos na memória em cada etapa.
3. **Características Centrais:** Para cada subcluster, o BIRCH mantém informações como a soma das distâncias quadradas de cada ponto ao centróide do subcluster e o número

de pontos pertencentes a ele. Essas características centrais são usadas para combinar subclusters durante a fase de construção hierárquica.

4. **Fase de Fusão Hierárquica:** À medida que novos dados são processados, os subclusters são mesclados em níveis mais altos da hierarquia, formando clusters maiores. Isso é feito com base nas características centrais dos subclusters.
5. **Agrupamento Final:** No final do processo, o BIRCH produz um conjunto de clusters hierárquicos ou um cluster global, dependendo das necessidades do usuário. A estrutura hierárquica pode ser explorada para identificar clusters em diferentes níveis de granularidade.

Principais vantagens do BIRCH:

- Eficiência em termos de uso de memória, tornando-o adequado para grandes conjuntos de dados.
- Tolerante a ruído, devido à sua estrutura hierárquica e capacidade de mesclar subclusters.
- Escalabilidade, permitindo que ele lide com grandes volumes de dados.

No entanto, o BIRCH também possui algumas limitações:

- Não é tão adequado para clusters de formas arbitrárias, como os algoritmos baseados em densidade, como o DBSCAN.
- A escolha dos parâmetros, como o fator de ramificação e o número máximo de subclusters por nó, pode afetar significativamente os resultados.

DEEP LEARNING NA CLUSTERIZAÇÃO

O Deep Learning envolve a construção e treinamento de redes neurais artificiais profundas para realizar tarefas complexas de aprendizado, como reconhecimento de padrões, classificação e clusterização. Nos últimos anos, o uso de técnicas de Deep Learning revolucionou a forma como a clusterização é realizada, permitindo uma abordagem mais flexível e precisa na identificação de padrões complexos nos dados. A principal vantagem do Deep Learning em relação a abordagens tradicionais de clusterização incluem [17]:

- **Extração automática de features**: As redes neurais profundas podem aprender features mais robustas e discriminativas diretamente dos dados brutos, sem necessidade de feature engineering manual.
- **Descoberta de representações não-lineares**: O deep learning é capaz de mapear os dados para espaços latentes não-lineares que facilitam a separação dos clusters.
- **Generalização aprimorada**: Os modelos profundos têm maior capacidade de generalizar para novos dados após o treinamento.

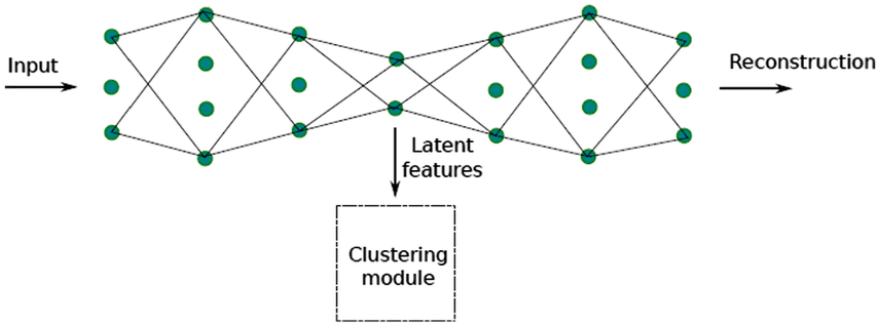
. Isso permite que modelos de Deep Learning identifiquem padrões abstratos e latentes nos dados, que podem ser cruciais para a formação de clusters significativos [18].

Deep Clustering Network (DCN)

Deep Clustering Network (DCN) [19] é uma abordagem avançada que combina técnicas de Deep Learning com a clusterização tradicional para realizar a tarefa de clusterização de dados de forma mais eficaz. O objetivo do DCN é aprender representações de dados de alta qualidade, que são então utilizadas para realizar a clusterização. Ao contrário de abordagens tradicionais, que muitas vezes requerem a definição manual de características (as features), o DCN permite que o próprio modelo aprenda representações relevantes durante o processo de treinamento.

A ideia central por trás do DCN é treinar uma rede neural profunda (geralmente uma autoencoder) para codificar os dados de entrada em um espaço latente de dimensão reduzida. Esse espaço latente captura as informações mais importantes dos dados e é projetado de tal forma que os pontos de dados similares são mapeados para regiões próximas no espaço latente. Uma vez que as representações latentes são aprendidas, técnicas de clusterização tradicionais, como o K-means, são aplicadas para agrupar os pontos de dados no espaço latente em clusters.

Figura 4 - DCN

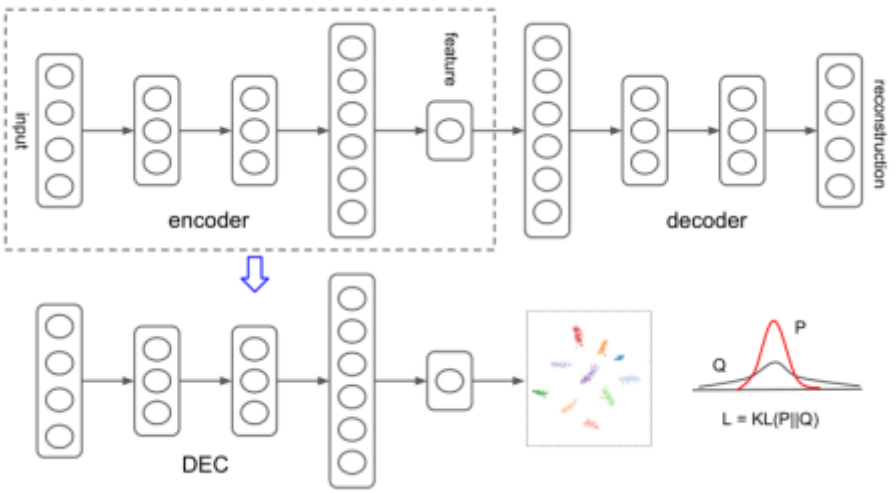


Deep Embedded Clustering (DEC)

No Deep Embedderd Clustering (DEC) [20], uma rede neural profunda é pré-treinada em uma tarefa não supervisionada, como um autoencoder, para aprender representações de alta qualidade dos dados de entrada. Isso ajuda a capturar as características importantes e latentes dos dados. Após o pré-treinamento, a camada de saída do autoencoder é removida e a rede neural é finamente ajustada usando um algoritmo de aprendizado de representações conjunto. Esse algoritmo busca otimizar as representações das amostras de forma que pontos de dados similares estejam mais próximos uns dos outros no espaço latente. Uma vez que as representações latentes são aprendidas, algoritmos de clusterização tradicionais, como o K-means, são aplicados no espaço latente para agrupar os pontos de dados em clusters.

Para melhor compreensão é recomendado ler o artigo:

Figura 5 - Diagrama do modelo DEC



Deep Adaptive Clustering (DAC)

O DAC é um método que utiliza um framework de classificação binária em pares. Dado dois pontos de dados de entrada, o modelo determina se os dados pertencem ao mesmo cluster ou não. Basicamente, há uma rede neural com uma ativação softmax que recebe um ponto de dados de entrada e gera um vetor com probabilidades do dado de entrada pertencer a

um conjunto específico de clusters. Dados dois pontos de entrada, o produto escalar dos resultados do modelo para ambos os dados é calculado. Quando as atribuições de cluster dos dois dados são diferentes, o produto escalar será zero, e para atribuições de cluster iguais, o produto escalar será um. Como o produto escalar é uma operação diferenciável, é possível treiná-lo com backpropagation usando rótulos de treinamento em pares.

Dado que os dados verdadeiros não estão disponíveis, as características da mesma rede são usadas para criar rótulos binários para o treinamento em pares. A distância cosseno entre as características dos dois pontos de dados é utilizada. Dado um par de entrada, se a distância cosseno for maior que um valor, então o par de entrada é considerado um par positivo (significando que ambos devem estar no mesmo cluster). Similarmente, se a distância cosseno for menor que um outro valor menor, então o par de entrada é considerado um par negativo (significando que ambos devem estar em clusters diferentes). Se a distância estiver entre os valores, o par é ignorado. Após obter os pares positivos e negativos, a perda em pares é minimizada.

Essa abordagem permite que o modelo aprenda representações adaptativas profundas para os dados, identificando relações entre pares de pontos de dados e, assim, melhorando a clusterização. O uso do framework de classificação binária em pares e a adaptação de limiares permitem uma abordagem mais flexível à clusterização em comparação com métodos tradicionais.

Para melhor compreensão leia o artigo presente em: https://openaccess.thecvf.com/content_iccv_2017/html/Chang_Deep_Adaptive_Image_ICCV_2017_paper.html

APLICAÇÃO DE CLUSTERIZAÇÃO EM PROBLEMAS REAIS

A clusterização desempenha um papel fundamental na vida real, permitindo a identificação de padrões ocultos e estruturas subjacentes em conjuntos de dados complexos. Na área de marketing, por exemplo, a segmentação de clientes com base em hábitos de compra semelhantes ajuda a personalizar campanhas e melhorar a satisfação do cliente. Na medicina, a clusterização de perfis genéticos pode levar à descoberta de subtipos de doenças, levando a tratamentos mais eficazes e personalizados. Na análise de redes sociais, a clusterização de redes de amizade pode revelar grupos de interesse e influenciadores. Em essência, a clusterização capacita diversas áreas, permitindo que informações valiosas sejam extraídas dos dados e aplicadas em decisões informadas e inovações.

Segue a referência de alguns artigos que utilizam os métodos abordados para analisar problemas reais:

- **Segmentação de clientes**: Identificar grupos de clientes com comportamentos ou necessidades semelhantes para personalizar campanhas de marketing e melhorar a experiência do usuário [21].
- **Detecção de fraude**: Detectar padrões anômalos em transações financeiras ou de telecomunicações pode ajudar a identificar tentativas de fraude [22].
- **Diagnóstico médico**: Agrupar imagens médicas ou registros de pacientes pode auxiliar no diagnóstico de doenças e na descoberta de subtipos [23].
- **Análise de redes sociais**: Encontrar comunidades em redes sociais baseado em interesses compartilhados para estudar interações e difusão de informações [24].
- **Bioinformática**: Clusterizar genes ou proteínas com funcionalidades similares para entender processos biológicos complexos [25].
- **Detecção de anomalias**: Identificar outliers que não se enquadram em nenhum cluster conhecido pode detectar falhas em equipamentos ou fraudes [26].

ARTIGOS QUE COMPARAM TÉCNICAS DE CLUSTERIZAÇÃO

Como já discutido, a clusterização é capaz de oferecer ao usuário insights importantes sobre os dados que ele analisa. Para isso é necessário conhecimento sobre como esses dados se estruturam para que se possa utilizar o método de clusterização mais eficaz para os dados. Seguem alguns artigos que discorrem sobre a comparação de alguns dos algoritmos citados:

Manning et al. [27] avaliaram o desempenho de 9 métodos populares de clusterização, incluindo k-means, spectral clustering e DBSCAN, em dados sintéticos com propriedades controladas. Eles analisaram critérios como qualidade da clusterização, escalabilidade e sensibilidade a parâmetros.

Charrad et al. [28] realizaram experimentos abrangentes em 8 conjuntos de dados de expressão gênica. Eles compararam 11 algoritmos, incluindo métodos tradicionais e baseados em grafos, com base em métricas de validação interna como o índice de Silhouette.

Sun et al. [29] focaram na avaliação de algoritmos para clusterização de dados de alto dimensionalidade. Eles utilizaram datasets texteis e de imagem para analisar o desempenho de 5 métodos de subespaço e projetaram um framework híbrido.

Min et al. [30] apresentam um survey focado em técnicas de clusterização baseadas em deep learning. Eles resumem as arquiteturas de diversos métodos modernos e discutem seus resultados em datasets de imagem e texto.

Em geral, os estudos concordam que não há um algoritmo universalmente superior, sendo crucial entender suas diferenças, pontos fortes e limitações para cada problema específico. Análises comparativas fornecem insights valiosos para orientar a seleção e aplicação de técnicas de clusterização.

UMA ANÁLISE COMPARATIVA

A escolha do método de clusterização mais eficiente e adequado desempenha um papel crucial na extração de informações significativas e na compreensão profunda de conjuntos de dados complexos. A clusterização é uma técnica fundamental no campo da análise de dados, pois agrupa objetos similares e revela padrões que podem passar despercebidos em uma análise superficial. No entanto, optar pelo método inadequado pode resultar em agrupamentos imprecisos e interpretações incorretas, comprometendo a utilidade das descobertas.

A eficiência do método de clusterização não está apenas relacionada à velocidade de processamento e à escalabilidade, mas também à capacidade de identificar os padrões intrínsecos presentes nos dados. Cada algoritmo de clusterização possui suposições, vantagens e limitações distintas. Portanto, a escolha do método deve ser guiada pelo conhecimento do domínio, pelas características dos dados e pelos objetivos da análise.

Um dos fatores mais importantes a considerar é a natureza dos dados. Alguns algoritmos, como o K-Means, funcionam bem com dados globulares e grupos de tamanho similar. Outros, como o DBSCAN, são mais adequados para clusters com formas e densidades variáveis. A compreensão das nuances dos dados é crucial para selecionar um método que capture efetivamente as estruturas de agrupamento.

Outra consideração é a interpretabilidade dos resultados. A clusterização deve ser um meio de simplificar a complexidade dos dados, tornando-os mais compreensíveis e acionáveis. Um método que produz agrupamentos mais coesos e distintos pode ser preferível quando a interpretação é vital. Nesse sentido, é essencial considerar a capacidade do método de criar clusters que se alinhem com os conhecimentos prévios ou intuições sobre os dados.

Por fim, vamos apresentar algumas comparações finais sobre os métodos de clustering apresentados a fim de auxiliar na tomada de decisão de que cluster deve ser utilizado:

Algoritmo	Prós	Contras
Kmeans	- Simples e eficiente	- Sensível à inicialização dos centróides
	- Escalável para grandes conjuntos de dados	- Pode convergir para mínimos locais
	- Converge rapidamente	- Requer definição prévia do número de clusters
	- Fácil de entender e implementar	- Pode não funcionar bem com clusters não convexos
Affinity Propagation	- Determina automaticamente o número de clusters	- Sensível aos parâmetros de damping e preference

	- Identifica automaticamente pontos exemplares	- Pode ser computacionalmente caro
	- Lida bem com formas e tamanhos diferentes	- Resultados podem variar com diferentes inicializações
Mean Shift	- Não requer especificação do número de clusters	- Pode ser computacionalmente caro e lento
	- Lida bem com clusters de diferentes formas	- Sensível à largura do kernel e outros parâmetros
	- Encontra clusters de formas irregulares	- Pode ficar preso em mínimos locais
Spectral Clustering	- Lida bem com clusters não convexos	- Requer conhecimento prévio do número de clusters
	- Pode capturar estruturas complexas nos dados	- Complexidade computacional pode ser alta
	- Efetivo em problemas de redução de dimensionalidade	- Sensível à escolha do método de similaridade
Agglomerative Clustering	- Pode lidar com diferentes tamanhos e formas de clusters	- Pode ser computacionalmente caro para grandes conjuntos de dados
	- Oferece opções de métodos de junção de clusters	- Pode não ser escalável para conjuntos de dados muito grandes
	- Pode capturar estruturas hierárquicas nos dados	- Sensível à escolha do método de junção de clusters
DBSCAN	- Lida com clusters de diferentes tamanhos e formas	- Sensível aos parâmetros de eps e min_pts
	- Identifica outliers como ruído	- Pode ser computacionalmente caro para conjuntos de dados densos
	- Não requer especificação do número de clusters	- Resultados podem variar com diferentes configurações de parâmetros
OPTICS	- Identifica automaticamente uma hierarquia de clusters	- Pode ser computacionalmente caro
	- Lida bem com diferentes tamanhos e formas de clusters	- Sensível aos parâmetros de eps e min_pts
	- Pode capturar clusters de diferentes densidades	- Necessita de pós-processamento para formar clusters finais

A escolha do método de clusterização apropriado depende das características dos dados e dos objetivos da análise. Diferentes algoritmos têm suas próprias forças e fraquezas, e podem ser mais adequados para cenários específicos. Aqui está uma análise das características de dados e algoritmos de clusterização correspondentes:

- **Dados com Número Conhecido de Clusters Convexos (K-Means)**: O algoritmo K-Means é particularmente eficaz quando o número de clusters é conhecido e os clusters têm formas aproximadamente convexas. Ele calcula centróides para cada cluster e atribui pontos ao cluster mais próximo ao centróide. É uma escolha sólida quando se busca agrupar dados em grupos bem definidos.
- **Dados com Número Desconhecido de Clusters (Affinity Propagation, Mean Shift)**: Quando o número de clusters não é conhecido, algoritmos como o Affinity Propagation e o Mean Shift podem ser úteis. O Affinity Propagation identifica "exemplares" dentro dos dados e atribui pontos aos clusters representados por esses exemplares. O Mean Shift é capaz de encontrar regiões de alta densidade que representam os centros dos clusters.
- **Dados com Clusters Não Convexos (Spectral Clustering, DBSCAN, OPTICS)**: Quando os clusters não têm formas convexas, algoritmos como o Spectral Clustering, DBSCAN e OPTICS são mais apropriados. O Spectral Clustering pode capturar padrões complexos usando representações de grafos. O DBSCAN e o OPTICS são sensíveis à densidade e podem identificar clusters de diferentes formas e tamanhos.
- **Dados com Outliers (DBSCAN, OPTICS)**: Algoritmos como o DBSCAN e o OPTICS são capazes de lidar bem com outliers, considerando-os como ruído ou pontos isolados. Eles ajudam a identificar regiões densas de pontos que formam clusters válidos, ignorando pontos isolados.

Algoritmo	Tipo	Número de Clusters	Tipos de Cluster	Escalabilidade	Outlier	Interpretabilidade
K-means	Particional	Necessita definir K	Convexos	Boa	Sensível	Alta
Affinity Propagation	Baseado em Message Passing	Descobre automaticamente	Formas variadas	Limitada	Robusto	Média
Mean Shift	Baseado em densidade	Descobre automaticamente	Não-convexos	Limitada	Robusto	Baixa
DBSCAN	Baseado em densidade	Descobre automaticamente	Formas variadas	Boa	Robusto	Alta
Spectral Clustering	Baseado em grafos	Necessita definir K	Não-convexos	Limitada	Moderadamente robusto	Baixa

BIRCH	Hierárquico	Necessita definir K	Formas variadas	Boa	Pouco robusto	Alta
OPTICS	Baseado em densidade	Descobre automaticamente	Densidades variáveis	Boa	Robusto	Média
DEC	Deep Learning	Necessita definir K	Complexos	Boa	Robusto	Baixa

Esta tabela fornece uma visão geral de propriedades importantes dos algoritmos como escalabilidade, flexibilidade na definição do número de clusters, capacidade de lidar com outliers, tipos de clusters suportados e facilidade de interpretação. Ela pode auxiliar na seleção do método mais adequado às características do problema e conjunto de dados. Porém, testes experimentais ainda são essenciais para validar o desempenho na prática.

CONCLUSÕES E SUGESTÕES DE COMPLEMENTAÇÕES PARA ESTUDOS

A clusterização é uma área fundamental da análise de dados, permitindo identificar padrões e estruturas ocultas em conjuntos de dados complexos. Este trabalho introduziu os principais conceitos, algoritmos e aplicações da clusterização, tanto métodos tradicionais como recentes baseados em deep learning.

Foi apresentada uma visão geral didática de diversas técnicas populares de clusterização, incluindo K-means, Affinity Propagation, Mean Shift, DBSCAN e Spectral Clustering. Também foram discutidos métodos modernos que empregam redes neurais profundas para aprimorar o processo de clusterização.

Além disso, foram exploradas aplicações relevantes da clusterização em problemas do mundo real e sugeridas expansões nas análises comparativas entre algoritmos. As seções reescritas buscaram trazer referências adicionais, links, explicações expandidas e uma visão crítica.

Como trabalhos futuros, seria valioso realizar uma avaliação experimental extensiva dos algoritmos em diversos tipos de dados para quantificar métricas de desempenho. Outras pesquisas podem se aprofundar no desenvolvimento de novos métodos híbridos entre técnicas tradicionais e deep learning. Há também espaço para aprimorar a interpretabilidade e visualização dos clusters descobertos. A clusterização continuará sendo uma área essencial para extrair insights a partir da análise de dados em diversas aplicações.

REFERÊNCIAS

[1] Alpaydin, E. (2020). Introduction to machine learning. MIT press.

[2] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.

[3] Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. ieee Computational intelligenCe magazine, 13(3), 55-75.

[4] Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement learning in robotics: A survey. The International Journal of Robotics Research, 32(11), 1238-1274.

[5] Bishop, C. M. (1995). Neural networks for pattern recognition. Oxford university press.

[6] Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). Foundations of machine learning. MIT press.

[7] Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. Science, 349(6245), 255-260.

[8] Ghahramani, Z. (2004). Unsupervised learning. Advanced lectures on machine learning, 72-112.

[9] Jain, A. K. (2010). Data clustering: 50 years beyond K-means. Pattern recognition letters, 31(8), 651-666.

[10] Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. ACM computing surveys (CSUR), 31(3), 264-323.

[11] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability (Vol. 1, No. 14, pp. 281-297).

[12] Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., & Wu, A. Y. (2002). An efficient k-means clustering algorithm: Analysis and implementation. IEEE transactions on pattern analysis and machine intelligence, 24(7), 881-892.

[13] Frey, B. J., & Dueck, D. (2007). Clustering by passing messages between data points. science, 315(5814), 972-976.

[14] Cheng, Y. (1995). Mean shift, mode seeking, and clustering. IEEE transactions on pattern analysis and machine intelligence, 17(8), 790-799.

[15] Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. Advances in neural information processing systems, 14.

[16] Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In Kdd (Vol. 96, No. 34, pp. 226-231).

[17] Min, E., Guo, X., Liu, Q., Zhang, G., Cui, J., & Long, J. (2018). A survey of clustering with deep learning: From the perspective of network architecture. IEEE Access, 6, 39501-39514.

[18] Aljalbout, E., Golkov, V., Siddiqui, Y., Strobel, M., & Cremers, D. (2018). Clustering with deep learning: Taxonomy and new methods. arXiv preprint arXiv:1801.07648.

[19] Yang, B., Fu, X., Sidiropoulos, N. D., & Hong, M. (2017). Towards k-means-friendly spaces: Simultaneous deep learning and clustering. International conference on machine learning (pp. 3861-3870). PMLR.

[20] Xie, J., Girshick, R., & Farhadi, A. (2016, June). Unsupervised deep embedding for clustering analysis. In International conference on machine learning (pp. 478-487). PMLR.

[21] Dolnicar, S. (2003). Using cluster analysis for market segmentation-typical misconceptions, established methodological weaknesses and some recommendations for improvement. Australasian Journal of Market Research, 11(2).

[22] Bolton, R. J., & Hand, D. J. (2002). Statistical fraud detection: A review. Statistical science, 235-249.

[23] Jain, A. K. (2010). Data clustering: 50 years beyond K-means. Pattern recognition letters, 31(8), 651-666.

[24] Fortunato, S. (2010). Community detection in graphs. Physics reports, 486(3-5), 75-174.

[25] Jiang, D., Tang, C., & Zhang, A. (2004). Cluster analysis for gene expression data: A survey. IEEE Transactions on knowledge and data engineering, 16(11), 1370-1386.

[26] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3), 1-58.

[27] Manning, C., Raghavan, P., & Schütze, H. (2008). Evaluation in information retrieval. Introduction to Information Retrieval, 151-175.

[28] Charrad, M., Ghazzali, N., Boiteau, V., & Niknafs, A. (2014). NbClust: An R package for determining the relevant number of clusters in a data set. Journal of Statistical Software,

61, 1-36.

[29] Sun, P., Choi, J. Y., Piao, M., Yoo, S. I., & Lee, K. H. (2018). A review of clustering methods for high-dimensional data. Cluster Computing, 1-18.

[30] Min, E., Guo, X., Liu, Q., Zhang, G., Cui, J., & Long, J. (2018). A survey of clustering with deep learning: From the perspective of network architecture. IEEE Access, 6, 39501-39514.

PSEUDO-CÓDIGOS

K-means

Input: S (Conjunto de dados), K (Número de clusters)
Output: clusters

1. Inicialize aleatoriamente k centróides
2. **Enquanto** condição de parada não for satisfeita:
3. Atribua cada ponto ao centróide mais próximo
4. Atualize os centróides para a média dos pontos em cada grupo
5. **Retorne** clusters

Affinity Propagation

Input: S (Conjunto de dados)
Output: clusters

1. Calcule as similaridades entre os pontos e monte a matriz de similaridade
2. Inicialize as matrizes de responsabilidades e disponibilidades
3. **Enquanto** condição de parada não for satisfeita:
4. Atualize a matriz de responsabilidade
5. Atualize a matriz de disponibilidades
6. Atualize os valores de exemplares (pontos de referência)
7. Identifique os clusters (exemplares)
8. **Retorne** clusters

Mean Shift

Input: S (Conjunto de dados), δ (Raio da banda)
Output: clusters

1. Inicialize os centróides como cópias dos pontos de dados
2. **Para** cada ponto de centróides:
3. **Enquanto** não convergiu:
4. Calcule o novo centróide como a média ponderada dos pontos dentro de δ
5. Verifique a convergência com base na distância entre o novo e o antigo centróide
6. Atualize o centróide para o novo centróide
7. Atribua cada ponto ao centróide mais próximo
8. **Retorne** clusters

Spectral Clustering

Input: S (Conjunto de dados), K (Número de clusters)
Output: clusters

1. Construa a matriz de similaridade
2. Calcule a matriz Laplaciana normalizada
3. Calcule os k principais autovetores da matriz Laplaciana normalizada
4. Selecione os autovetores
5. Execute o algoritmo de clustering (por exemplo, K-Means) nos autovetores
6. **Retorne** clusters

Agglomerative Clustering (Ward, Single, Average, Complete)

Input: S (Conjunto de dados), K (Número de clusters)
Output: clusters

```
1. Inicialize cada ponto de dados como seu próprio cluster
2. Enquanto o número de clusters > k:
3.     Encontre os dois clusters mais próximos utilizando os critérios de
   mesclagem
4.     Combine os dois clusters em um novo cluster
5. Retorne clusters
```

DBSCAN

Input: S (Conjunto de dados), min_pts (Mínimo de pontos), eps (Raio da vizinhança)
Output: clusters

```
1. Para cada ponto no conjunto de dados:
2.     Se o ponto não tiver sido visitado:
3.          Marque o ponto como visitado
4.          Se o número de vizinhos >= min_pts:
5.               Comece um novo cluster com o ponto atual
6.               Para cada vizinho em vizinhos:
7.                   Se o vizinho não estiver marcado como visitado:
8.                        Marque o vizinho como visitado
9.                        Expanda o cluster recursivamente
10.         Senão
11.              Marque o ponto como ruído
12. Retorne clusters
```

OPTICS

Input: S (Conjunto de dados), eps (Raio da vizinhança), min_amostras (Mínimo de pontos)
Output: clusters

```
1. Crie um grafo de Vizinhos Mais Próximos (neighbors_graph) com raio = eps e
   min_amostras = min_amostras
2. Para cada ponto em dados:
3.     Se o ponto não foi processado:
4.         Encontre ordered_points identificando vizinhos dentro da distância
   eps
5.         Marque o ponto como processado
6.         Calcule a distância central (core distance) para o ponto e defina-a
   em core_distances
7.         Se a distância central for menor ou igual a eps:
8.              Expanda o cluster
9.
10. Retorne clusters
```