UNIVERSIDADE FEDERAL DE LAVRAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Aluno: Michel Alexandrino de Souza **Matrícula: 202111084**
Aluno: João Pedro Alves Carneiro Valadão **Matrícula: 202020295**

GCC128 – Inteligência Artificial **Prof. Ahmed Ali Abdalla Esmin**

Relatório Técnico – K-means

1. Definição e apresentação da base de dados

O conjunto de dados Iris, introduzido por Ronald Fisher em 1936, constitui um benchmark clássico em machine learning, frequentemente utilizado para tarefas de classificação supervisionada. Composto por 150 amostras igualmente distribuídas entre três espécies do gênero Iris (setosa, versicolor e virginica), o dataset apresenta quatro atributos numéricos contínuos: comprimento e largura da sépala (em cm), e comprimento e largura da pétala (em cm). Cada instância está perfeitamente categorizada, sem valores faltantes ou inconsistentes, o que o torna ideal para prototipagem de algoritmos. A relevância científica desta base reside em suas propriedades estatísticas bem comportadas e na separabilidade parcial das classes. Enquanto Iris setosa é linearmente separável das demais, versicolor e virginica apresentam regiões de sobreposição em seu espaço de características, criando um desafio interessante para classificadores. A distribuição balanceada (50 amostras por classe) elimina a necessidade de técnicas de reamostragem, permitindo uma análise mais direta dos agrupamentos gerados. A abordagem de agrupamento utilizada neste trabalho foi o algoritmo K-means.

O K-means é um método de aprendizado não supervisionado amplamente empregado em tarefas de clusterização, cujo objetivo é particionar os dados em k grupos distintos com base na similaridade entre as instâncias. Inicialmente, k centróides são definidos de forma aleatória. Em seguida, cada amostra do conjunto é atribuída ao centróide mais próximo, com base em uma métrica de distância — neste caso, a distância euclidiana, definida por:

$$d\,(p,q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

onde p e q são vetores n-dimensionais que representam duas amostras, e p_i, q_i são os valores dos atributos na i-ésima dimensão. Após essa etapa de atribuição, os centróides são

recalculados como a média das amostras pertencentes a cada grupo. O processo é repetido iterativamente até que os centróides se estabilizem, ou seja, até que a configuração dos clusters não sofra alterações significativas entre as iterações. No uso do conjunto de dados Iris, a variável alvo foi desconsiderada para respeitar o caráter não supervisionado do algoritmo K-means. Foram realizados experimentos com $k = 3$ e $k = 5$, permitindo observar como as amostras se agrupam naturalmente no espaço de características. A qualidade dos agrupamentos foi avaliada por meio do *silhouette score*, que considera a coesão interna e a separação entre os clusters. Para facilitar a visualização, aplicou-se a técnica de PCA com 1 e 2 componentes principais, permitindo representar os clusters e seus centróides em dimensões reduzidas.

2. Comparação entre as aplicações

Neste trabalho, foram implementadas duas abordagens distintas para o algoritmo K-means: uma versão desenvolvida manualmente (hardcore), sem o uso de bibliotecas específicas para clusterização, e outra utilizando a implementação disponível na biblioteca scikit-learn. Ambas as versões foram aplicadas ao conjunto de dados Iris (sem considerar a variável alvo), com o objetivo de comparar o desempenho, a qualidade dos agrupamentos e as limitações de cada abordagem. A avaliação foi realizada por meio da métrica *silhouette score*, permitindo medir a coesão e separação dos clusters formados. Além disso, utilizou-se a técnica de redução de dimensionalidade PCA, com 1 e 2 componentes principais, a fim de possibilitar a visualização gráfica dos agrupamentos e dos centróides em um espaço de menor dimensão.

Implementação Hardcore

A implementação manual do algoritmo K-means, chamada KMeansHardcore, foi desenvolvida para demonstrar o funcionamento interno do algoritmo. Ela utiliza inicialização aleatória dos centróides, cálculo de distâncias euclidianas e atualização iterativa dos centróides até a convergência ou até atingir o número máximo de iterações. Embora funcional, essa abordagem não possui otimizações avançadas, como inicialização inteligente (k-means++) ou paralelismo, o que pode impactar o desempenho em grandes conjuntos de dados.

Implementação com Scikit-learn

A implementação com scikit-learn utiliza a classe KMeans, que é altamente otimizada e inclui recursos como inicialização k-means++, múltiplas inicializações (n_init) e suporte a grandes conjuntos de dados. Essa abordagem é mais eficiente e robusta, além de ser mais fácil de usar, já que encapsula toda a lógica do algoritmo em métodos simples como fit e predict.

Comparação de Desempenho

O desempenho foi avaliado em termos de tempo de execução. A implementação com scikit-learn foi significativamente mais rápida devido às otimizações internas e ao uso de múltiplas inicializações para encontrar melhores soluções. Por outro lado, a implementação hardcore, sendo mais básica, apresentou tempos de execução mais elevados, especialmente para valores maiores de k.

Comparação de Métricas

Ambas as implementações foram avaliadas usando o *silhouette score*, que mede a qualidade dos clusters gerados. Os resultados foram semelhantes para ambas as abordagens, indicando que a implementação hardcore é funcional e gera clusters de qualidade comparável à versão otimizada do scikit-learn. No entanto, a implementação com scikit-learn apresentou maior consistência devido ao uso de inicialização k-means++.

3. Resultados e limitações

Resultados

Os resultados mostraram que o número de clusters (k) influencia diretamente a qualidade dos agrupamentos. Para o dataset Iris, o valor $k = 3$ apresentou o melhor *silhouette score*, refletindo a divisão natural do conjunto de dados em três classes. Além disso, as matrizes de confusão e os gráficos PCA ajudaram a visualizar os clusters e a comparar os resultados com as classes reais.

Limitações

A implementação hardcore apresenta limitações em termos de desempenho e escalabilidade, sendo inadequada para grandes conjuntos de dados. Além disso, a inicialização aleatória dos centróides pode levar a resultados inconsistentes. Já a implementação com scikit-learn é limitada pela necessidade de ajustar manualmente o número de clusters (k), o que pode ser desafiador em cenários reais.

Considerações Finais

O trabalho demonstrou a eficácia do algoritmo K-means para agrupamento de dados e destacou as vantagens de usar bibliotecas otimizadas como scikit-learn. A implementação hardcore foi útil para entender os fundamentos do algoritmo, enquanto a versão com scikit-learn mostrou-se mais prática e eficiente para aplicações reais. No futuro, explorar métodos para determinar automaticamente o número ideal de clusters poderia melhorar ainda mais os resultados.