Análise de dados de expressão gênica

Clarissa Simoyama David†, Gustavo Torres Custódio† e Saulo Marx de Paiva†

†*Universidade Federal do ABC*
{*clarissa.david,gustavo.torres,saulo.marx*}*@aluno.ufabc.edu.br*

***Resumo*—Este projeto tem como intuito implementar e avaliar os algorítmos de k vizinhos mais próximos (KNN), k-médias (*K-Means*) e *Naive Bayes*, que são amplamente utilizadas em bioinformática para melhor classificar sequências genéticas e realizar comparações entre estas sequências. O tema escolhido foi o item 2, análise de dados de expressão gênica, como projeto final para a disciplina Introdução à Bioinformática.**

1. Introdução

Este projeto para a disciplina Introdução à Bioinformática consiste na implementação de uma extensão de uma das atividades práticas realizadas durante o quadrimestre. Dentre as atividades propostas, a que fora escolhida para a implementação fora a atividade do item 2, análise de dados de expressão gênica, que consiste em, com a base de dados *golub*, em formato csv, rodar algoritmos de classificação: o *K-Means*, o KNN e um de nossa escolha, variando seus parâmetros e reportando seus resultados.

As ferramentas utilizadas para a implementação deste projeto foram a linguagem de programação Python, com as bibliotecas *Numpy* e *Scikit-learn*, além dos conteúdos vistos em sala de aula.

O texto está organizado da seguinte forma: na Seção 2 os algoritmos de classificação são detalhados, fornecendo uma introdução sobre os suas técnicas de metodologia. A Seção 3 detalha os métodos utilizados na condução dos experimentos. Na Seção 4 são mostrados os resultados obtidos nos experimentos realizados. A Seção 5 apresenta as conclusões deste projeto e trabalhos futuros.

2. Referencial Teórico

Esta seção explica detalhadamente os algoritmos *KNN*, *K-Means* e *Naive Bayes*, além de fornecer uma introdução sobre alguns conceitos básicos sobre agrupamento de dados e classificação, utilizados para encontrar informações importantes sobre conjuntos de dados.

2.1. Apredizagem de Máquina e Técnicas de Agrupamento

Aprendizado de Máquina (AM) é uma técnica de criar algorítimos capazes de melhorar seu desempenho de acordo com a experiência adquerida[4]. Um dos problemas resolvidos pelo AM é o de agrupamento onde existe um espaço de características e com isso é possível criar grupos de características similares. Logo ao implementar um algoritmo de agrupamento simplesmente entramos com uma base de dados mais simples para o treino para gerar experiência e depois são adicionados os demais dados. Os algoritmos de KNN e K Means apresentados a seguir usam essa abordagem.

2.2. KNN

O Algoritmo dos k vizinhos mais próximos [1] (do inglês *K Nearest Neighbor*, KNN) é uma técnica de Aprendizado de Máquinas que representa os objetos como pontos no espaço. A classificação de cada elemento no espaço é feita de acordo com a distância entre os k objetos mais próximos do elementos. Isso é possível pois é realizado um treinamento com uma base teste e com uma base de treinamento. Normalmente a métrica utilizada para calcular a distância entre os objetos é a distância euclidiana, vista na Equação 1, sendo n o número de dimensões, i a coordenada atual calculada, x_i e y_i os valores das coordenadas dos dois pontos.

$$distancia = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{1}$$

Uma decisão importante é a escolha do número k de vizinhos, como visto na Figura 1, pois se o k for um número pequeno pode não considerar elementos próximos e importantes na classificação. Por outro lado, se k for um número grande, pode haver problemas em definir qual grupo o elemento pertence, pois a comparação é feita com diversos vizinhos diferentes, além de possuir um custo computacional maior.

Em bioinformática, a técnica de KNN pode ser utilizada para ajudar em problemas complexos como Busca por Sinal - *Splicing* [3], auxílio na descoberta de diagnósticos, entre outros. É um algoritmo de treinamento simples e incremental.

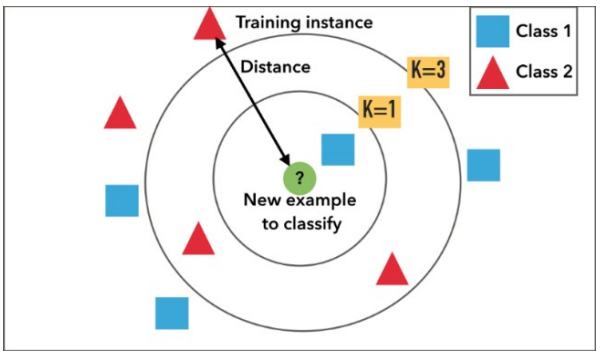

Figura 1. Exemplo do algoritmo KNN visto graficamente.

2.3. K-Means

O algoritmo *k-means* (ou k-médias) [2] pertence à família de técnicas de agrupamento (mais especificamente, *hard clustering*), com o objetivo de particionar os objetos dentre k grupos diferentes, cada um com seu centróide, no qual cada objeto pertence ao grupo mais próximo da média. É amplamente utilizado devido a sua simplicidade, interpretabilidade e eficiência computacional.

Para sua funcionalidade, primeiramente são selecionados a quantidade de grupos para uma base de dados (sendo este o valor de k) e inicializado os centróides de forma aleatória. Em seguida, é feito o cálculo da dissimilaridades entre os objetos da base de dados e os centróides, encontrando grupos iniciais pela regra do vizinho mais próximo. Com isso, é atualizado os valores dos centróides dos grupos, sendo necessária novamente a realização do cálculo das dissimilaridades entre objetos e centróides, e assim por diante até haver uma convergência, como pode ser visto na Figura 2.

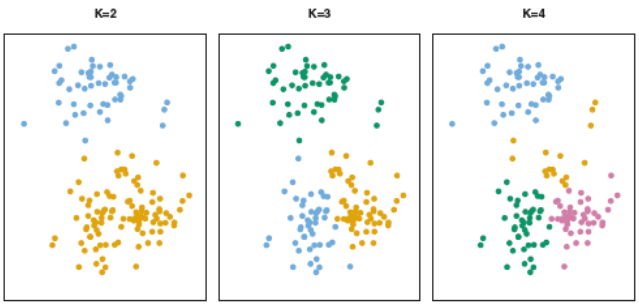

Figura 2. Exemplo do algoritmo *K-Means* visto graficamente.

Geralmente, a escolha da quantidade de grupos para a implementação do *k-means* varia entre uma faixa a partir do número de classes da base de dados escolhida até o piso da raiz quadrada da quantidade de objetos pertencentes à essa base, sendo a quantidade de grupos podendo ser variável de uma rodada para a outra. A proximidade pode ser calcula de diversas formas, sendo algumas delas a distância euclidiana, correlação, entre outros.

2.4. Naive Bayes

O Naive Bayes é um modelo de probabilidade condicional. Dado uma instância de dados $x = (x_1, x_2, ..., x_n)$, assumimos que cada dimensão de x é totalmente independente da outra. o Naive Bayes calcula a probabilidade de um exemplo de dados x pertencer a uma classe C ($P(C|x)$). Usando o teorema de Bayes, essa probabilidade pode ser reescrita como:

$$P(C|x) = \frac{P(x|C).P(C)}{P(x)} \tag{2}$$

O Naive Bayes considera que cada atributo x_i da entrada de dados é independente, dessa forma, podemos reescrever a fórmula anterior como:

$$P(C|x) = \frac{P(x_1|C).P(x_2|C)...P(x_n|C).P(C)}{P(x_1|C).P(x_2|C)...P(x_n|C)} \tag{3}$$

Para cada classe possível do problema, há uma probabilidade $P(C|x)$ diferente. Para definir a qual classe o exemplo pertence, escolhemos a classe com maior probabilidade.

3. Experimentos

Nesta seção serão explicados como os experimentos foram realizados e todos os passos necessários para a reprodução dos mesmos.

3.1. Configuração do KNN

Para os valores de k escolhidos na implementação do algoritmo KNN, foram variados do valor 1 até o valor 5, nos dando a acurácia de predição para cada número de k. Neste caso, a base de dados fora separada entre teste e treino e o resultado fora computado graficamente.

3.2. Configuração do K-Means

Para o valor de grupos escolhidos na implementação do algoritmo *K-Means*, foram testados valores de 2 (que é originalmente a quantidade de classes que a base de dados possui) até 6 grupos (a base de dados possui 38 objetos, portanto o valor máximo ideal de grupos para ser testado seria de valor 6).

3.3. Configuração do Naive Bayes

O Naive Bayes utilizado foi o gaussiano. A característica desse tipo de Naive Bayes é que ele usa uma função gaussiana para calcular probabilidades (4).

$$p(x = k|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{\frac{-(k-\mu_c)^2}{2\sigma_c^2}} \tag{4}$$

k: valor de x

μ_c: média dos valores de x associados com a classe c

σ_c: variância de valores de x associados com a classe c

3.4. *Framework* Desenvolvido

Todos os algoritmos desenvolvidos foram implementados na linguagem de programação *Python* estavam presentes na biblioteca *scikit-learn*, no qual possui diversos algoritmos para aprendizado de máquinas em Python.

Também foi utilizado para todos os algoritmos a mesma base de dados, chamada *golub*, em formato de arquivo *csv*. Ela possui 38 objetos e 2 classes. Todos os algoritmos podem ser rodados via terminal ou com auxília de alguma IDE (na disciplina fora utilizada a IDE *Spyder*), dando os resultados finais, como visto na Figura 3.

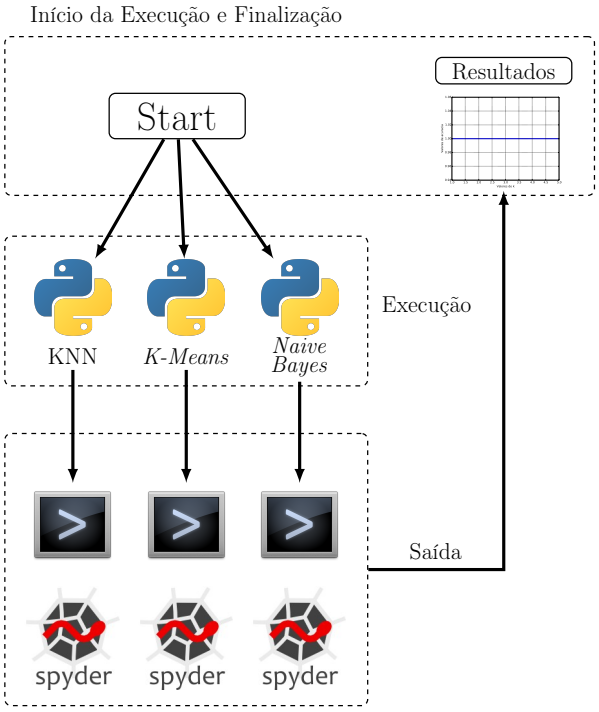

Figura 3. Fluxograma representando as fases/componentes do *framework* desenvolvido.

O código-fonte do *framework* está disponível no repositório GitHub[1].

3.5. Avaliação dos Resultados

Para cada algoritmo implementado neste projeto, foram escolhidos diferentes maneiras para as avaliações dos resultados.

Para o algoritmo KNN, fora pedido para reportar a acurácia de predição. Para isto, fora utilizado uma função da biblioteca *scikit-learn*, do pacote chamado *sklearn.metrics*, chamada *accuracy_score*, na qual através do teste e do que fora predito, ela devolve o valor da acurácia de predição. Com estes valores, fora plotado um gráfico para melhor visualização dos resultados.

1. https://github.com/ClaDavid/AMBioinformatica

Para o algoritmo K Means, fora pedido para tentar identificar o número de sub grupos de câncer. Para isto, foram utilizadas duas funções da biblioteca *scikit-learn*, do pacote *sklearn.metrics*, chamadas *accuracy_silhouette_samples* e *accuracy_silhouette_score*[6], na qual através de gráficos de distribuição das distâncias dos grupos de teste e a pontuação da silhueta [5], e possível determinar quais possíveis valores de k são ideais.

Para o Naive Bayes, separamos os dados de entrada em dados de treinamento e dados de teste. Utilizamos 2/3 dos dados como dados de treinamento e o 1/3 restantes foram usados como dados de teste, ou seja, dados usados para calcular a acurácia da predição.

4. Resultados

Para os resultados do algoritmo KNN, a figura 4 mostra a acurácia da predição pela variação do número de vizinhos k. Por ser uma base de dados relativamente pequena, logo com o valor de $k = 1$ fora obtido 100% de acurácia de predição.

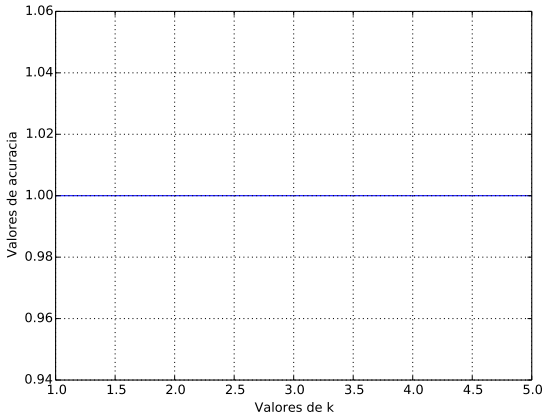

Figura 4. Valores da quantidade de k vizinhos x Valores de acurácia

No algoritmo *K-Means*, para os resultados das análise do número ideal de grupos, as figuras 5 à 9 representam a distribuição das distâncias médias entre os grupos.

Notamos que nas figuras 8 e 9 exite distribuições menores do que a média, logo os valores de 5 e 6 não são interessantes para k. Para k 2 ou 3 existem grupos que são mais concentrados do que outros (em 5 seria 1 e 6 em seria 2). O que possui valores dentro da média e uma distribuição de elementos melhor seria o k igual a 4 visto em 7.

O Naive Bayes conseguiu fazer a predição correta de todos os exemplos na configuração em que usamos 2/3 dos dados para treinamento e 1/3 para teste. Foi realizado também um experimento usando 1/3 dos dados para treinamento e 2/3 para teste. Nessa configuração, a taxa de acertos foi de 0.68 como visto em 10.

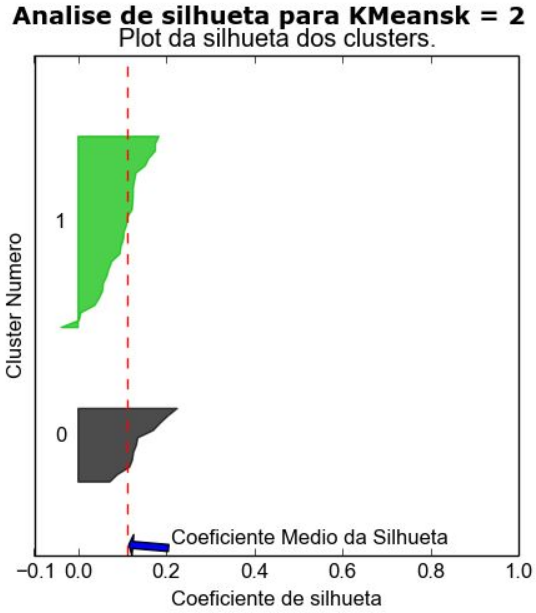

Figura 5. Comparação das silhuetas dos grupos com $k = 2$

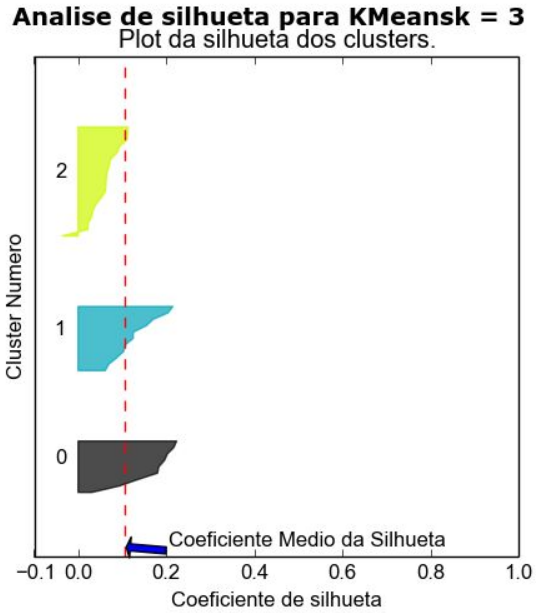

Figura 6. Comparação das silhuetas dos grupos com $k = 3$

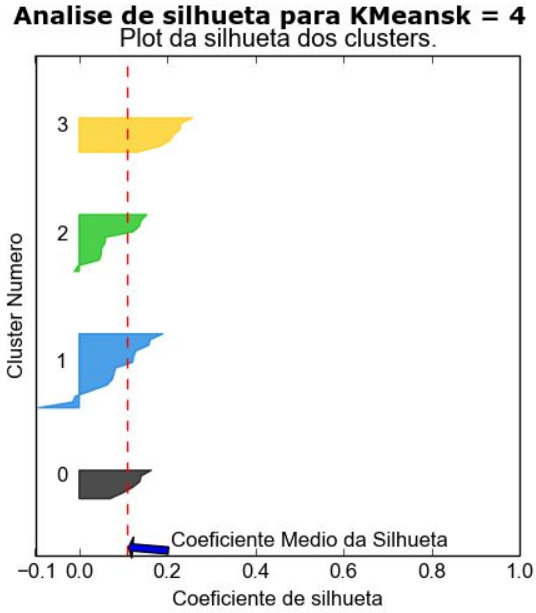

Figura 7. Comparação das silhuetas dos grupos com $k = 4$

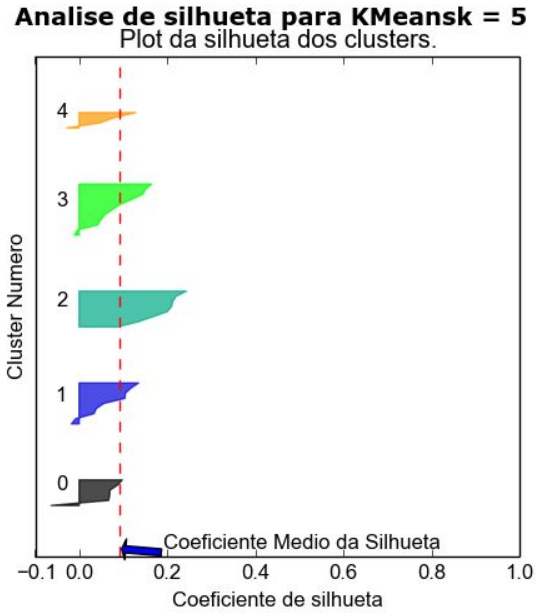

Figura 8. Comparação das silhuetas dos grupos com $k = 5$

5. Conclusão e Trabalhos Futuros

A aplicação de técnicas de classificação na bioinformática pode auxiliar no desenvolvimento e crescimento nesta vasta área. Um profissional que consiga entender ambos os lados, tanto da área bilógica quanto desta área computacional, é fundamental para o avanço em campos como análise dos grupos de dados genéticos que existem tanto na literatura científica quanto nos dados experimentais.

Pode-se observar que mesmo com a implementação simples de algumas técnicas, muitas informações podem ser colhidas e podem dar resultados sobre base de dados importantes, como a utilizada na realização deste projeto. Todos os resultados obtidos foram de acordo com o que era esperado, e as classificações se mostraram corretas.

As sugestões que foram dadas de extensões foram implementadas, e como trabalhos futuros está a aplicação destas técnicas em uma base de dados maior (para realizar mais testes sobre a acurácia das técnicas).

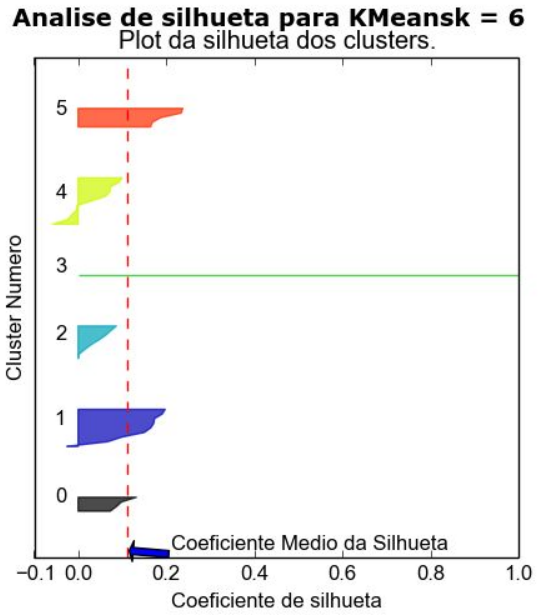

Figura 9. Comparação das silhuetas dos grupos com $k = 6$

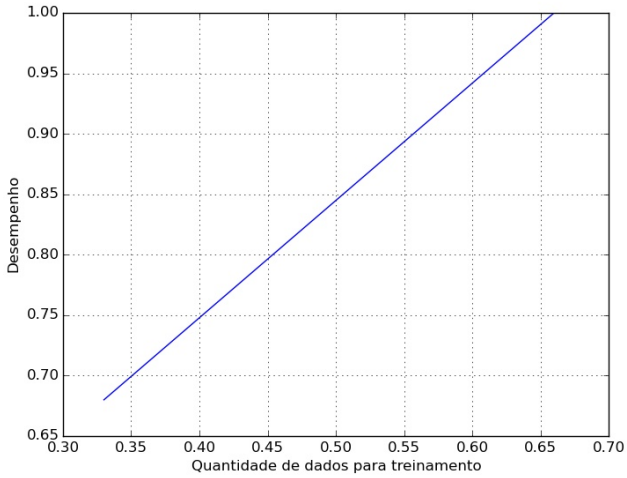

Figura 10. Desempenho do Naive Bayes de acordo com o tamanho dos dados de treinamento.

Referências

[1] Thomas Cover e Peter Hart. "Nearest neighbor pattern classification". Em: *IEEE transactions on information theory* 13.1 (1967), pp. 21–27.

[2] John A Hartigan e Manchek A Wong. "Algorithm AS 136: A k-means clustering algorithm". Em: *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28.1 (1979), pp. 100–108.

[3] A Lapedes et al. *Application of neural networks and other machine learning algorithms to DNA sequence analysis*. Rel. téc. Los Alamos National Lab., NM (USA), 1988.

[4] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[5] Peter J Rousseeuw. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis". Em: *Journal of computational and applied mathematics* 20 (1987), pp. 53–65.

[6] *Selecting the number of clusters with silhouette analysis on KMeans clustering*. http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html. Accessed: 2016-08-28.