Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Algoritmos para Bioinformática I - Primeiro Semestre 2020

Trabalho Prático 4 - Mineração de Dados utilizando álgebra linear

Aluno: Guilherme de Abreu Lima Buitrago Miranda - Matrícula: 2018054788

1. Introdução

Recentemente, com a pandemia da covid-19, causada pelo vírus SARS-CoV2, muito se tem acompanhado nos veículos de comunicação sobre a tentativa dos cientistas em encontrar um medicamento que seja capaz de curar ou, pelo menos, aumentar a taxa dos recuperados da doença. Em particular, várias tentativas de uso *off label* de drogas farmacêuticas foram feitas, ou seja, utilização de fármacos que não foram originalmente desenvolvidos para a covid-19. Contudo, é interessante questionar: como os cientistas decidem quais remédios irão tentar utilizar para a busca de uma nova cura?

Além de se atentar às características biológicas do patógeno, os cientistas têm a possibilidade de utilizar a bioinformática para, por exemplo, encontrar, em outro vírus, uma proteína parecida com a do vírus estudado. Assim, se um dado remédio é útil para curar infecções com o primeiro vírus, há boas chances desse mesmo remédio ajudar a controlar a infeção do novo vírus estudado.

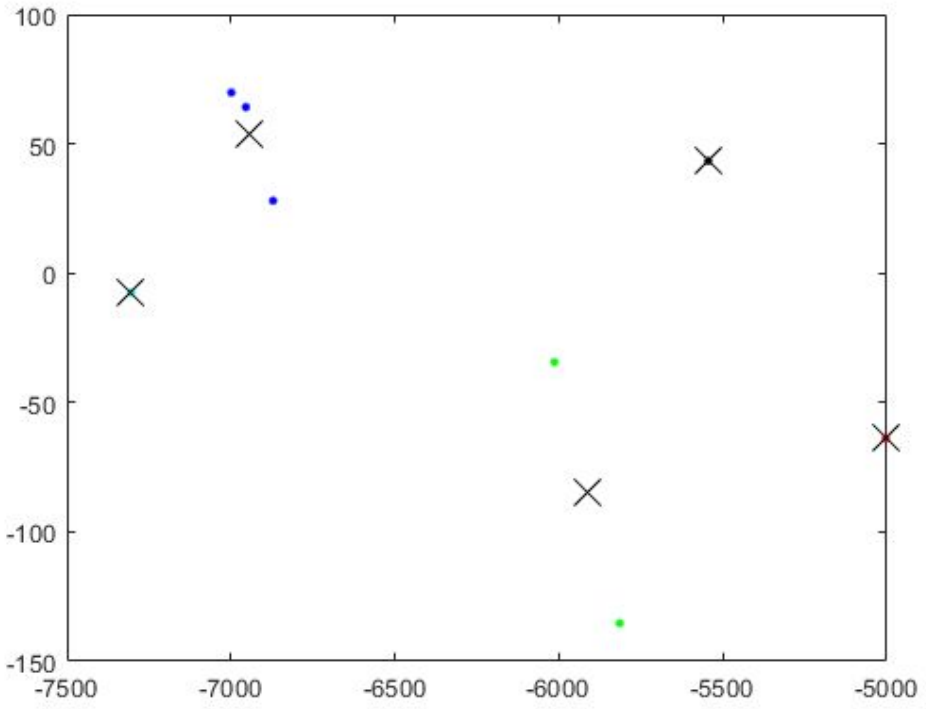
Dessa forma, este trabalho busca estudar algumas espécies de proteína, utilizando os já estudados conceitos de mineração de dados e álgebra linear. Assim, o script desenvolvido determina o grau de semelhança das proteínas e constrói uma árvore hierárquica associada, assim como separa as mesmas em grupos (*clusters*). Dessa forma, em primeiro lugar, explica-se as entradas esperadas e as saídas obtidas. Em seguida, discute-se sentido de cada uma das informações geradas e os métodos utilizados pelo algoritmo. Por fim, debate-se as conclusões alcançadas a partir do trabalho.

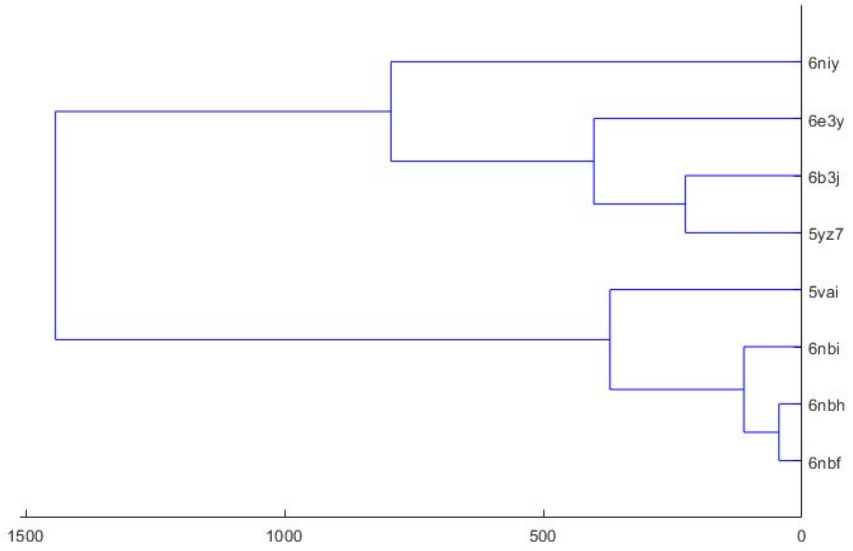
2. Entradas e Saídas

Como entrada para o script do trabalho trabalho, é esperada uma série de arquivos .pdb com o mesmo chain ID. Neste trabalho, foram utilizados os seguinte arquivos: 5uz7.pdb, 5vai.pdb, 6b3j.pdb, 6e3y.pdb, 6nbf.pdb, 6nbh.pdb, 6nb.pdb e 6niy.pdb.

Após o processamento dos arquivos e a aplicação das técnicas de mineração de dados e álgebra linear, que serão melhor detalhadas na seção 3, obtém-se duas saídas: uma visualização em duas dimensões da separação dos grupos realizada pelo algoritmo kmeans e uma árvore hierárquica relativa às proteínas analisadas.

Abaixo, pode-se observar tais resultados, sendo o primeiro a separação do kmeans em 5 grupos e suas respectivas centróides, e o segundo a árvore hierárquica das proteínas utilizadas na entrada para o script.





Após a observação dos resultados, é interessante observar a grande semelhança da árvore obtida com aquela presente no *site* pfam. Conforme desejado, tem-se que as estruturas 6nbf, 6nbh e 6nbi estão separadas apenas no final, pois são oriundas da mesma proteína - PTH1R_HUMAN. Além disso, as mesmas estão consideravelmente distantes da estrutura 5uz7, presente em uma proteína diferente, a CALCR_HUMAN.

3. Metodologia

Após a obtenção da matriz A utilizando o script TP4a nos moldes do vector space model, realiza-se a decomposição por valores singulares (SVD), com o intuito de cotejar o número de grupos e visualizar o domínio do problema.

Posteriormente, com base nos dados gerados, estima-se a quantidade de padrões que deseja-se utilizar e a quantidade de grupos. Em seguida, faz-se a separação dos mesmos utilizando o algoritmo *kmeans.* A partir dos dados já obtidos pela decomposição SVD, o algoritmo em questão separa as proteínas observando as semelhanças e as diferenças entre suas respectivas representações no vector space model.

Em especial, a separação dos grupos utilizando o algoritmo *kmeans* mostrou-se muito relevante, visto que a mesma coincide com aquela esperada quando observa-se o *site* pfam. Conforme desejado, estruturas pertencentes a

diferentes proteínas são agrupadas separadamente, enquanto estruturas pertencentes às mesmas proteínas são agrupadas juntas.

Por último, é construída uma árvore hierárquica associada às entidades representadas pela matriz A. Nela, em consonância com o algoritmo *kmeans,* quanto mais parecidas são as estruturas, mais curta é a separação. Em contrapartida, quanto mais diferentes são as estruturas, mais cedo elas são separadas.

Em concordância com o destacado na seção 2, a árvore construída foi muito satisfatória no sentido que é semelhante à árvore filogenética apresentada pelo *site* pfam, como desejado.

4. Conclusões

O trabalho mostrou-se muito proveitoso para que se observasse como a técnica de decomposição SVD, combinada com o algoritmo *kmeans* e a construção de árvores associadas às entidades, se comporta, quando aplicada ao estudo das estruturas das proteínas.

Em particular, é extremamente pertinente poder observar que, assim como certos padrões se repetem nas estruturas compostas por bases nitrogenadas (DNAs e RNAs), têm-se algo parecido com as proteínas. Evidentemente, no lugar das bases nitrogenadas, entram os aminoácidos. Assim, na maior parte dos casos, independente da proteína estudada, tem-se cinco valores mais importantes após a decomposição SVD.

Dessa forma, o trabalho nos permite devanear a respeito de outras possíveis aplicações para as técnicas de mineração de dados utilizando álgebra linear no estudo das proteínas, como a descoberta de novos fármacos, por exemplo.

5. Referências Bibliográficas

- Pfam database - https://pfam.xfam.org/
- Protein Data Bank - https://www.rcsb.org/
- Kmeans - Help Center - https://www.mathworks.com/help/stats/kmeans.html
- Dendrogram - Help Centeer - https://www.mathworks.com/help/stats/dendrogram.html

6. Anexos

Além dos scripts TP4a.m e LeiaCoordenadasPDB.m fornecidos, foram utilizados, como anexos, os arquivos 5uz7.pdb, 5vai.pdb, 6b3j.pdb, 6e3y.pdb, 6nbf.pdb, 6nbh.pdb, 6nb.pdb e 6niy.pdb, disponíveis no site Protein Data Bank.

Posteriormente, foram executados 8 vezes os seguintes comandos:

```
TP4a;
A = [A s(1:10)];
```

A única diferença entre as execuções é na edição do arquivo TP4a.m, em que a variável Arq era modificada a depender do arquivo .pdb a ser lido. Finalmente, obtém-se uma matriz A de dimensões 10x8. Em seguida, tem-se os seguintes passos:

```
[U, S, V] = svd(A);
Aux = S*V';
x = Aux(1, :);
y = Aux(2, :);
plot(x, y, '*')
```

```
Aux_kmeans = transpose(Aux(1:5, :));
[idx, C] = kmeans(Aux_kmeans, 5);
figure
plot(Aux_kmeans(idx==1,1),Aux_kmeans(idx==1,2),'r.','MarkerSize',1
0)
hold on
plot(Aux_kmeans(idx==2,1),Aux_kmeans(idx==2,2),'b.','MarkerSize',1
0)
plot(Aux_kmeans(idx==3,1),Aux_kmeans(idx==3,2),'g.','MarkerSize',1
0)
plot(Aux_kmeans(idx==4,1),Aux_kmeans(idx==4,2),'c.','MarkerSize',1
0)
plot(Aux_kmeans(idx==5,1),Aux_kmeans(idx==5,2),'k.','MarkerSize',1
0)
plot(C(:,1),C(:,2),'kx', 'MarkerSize',15)
```

```
arvore = linkage(transpose(Aux (1:5, :)), 'average');
```

```
labels = {'5yz7', '5vai', '6b3j', '6e3y', '6nbf', '6nbh', '6nbi',
'6niy'}
dendrogram(arvore, 'Orientation', 'left', 'Labels', labels);
```