Universidade do Estado de Santa Catarina
Centro de Ciências Tecnológicas
Departamento de Ciência da Computação
Bacharelado em Ciência da Computação

Alunos: Adriano Pereira Silva, Herton Silveira e Victor Perim

Grafo Íris

Teoria dos Grafos
Prof. GILMARIO BARBOSA DOS SANTOS

Joinville/2024

Introdução

Este trabalho da disciplina de Teoria dos Grafos aborda a implementação e avaliação de um modelo de clustering baseado em grafos, utilizando a linguagem C para processar dados da base Iris. Esta base contém características morfológicas de três espécies do gênero flor íris (setosa, versicolor e virginica), e o principal desafio é agrupar as observações em classes distintas, considerando a complexidade de separar corretamente as espécies através de técnicas de análise de grafos. Além da implementação de algoritmos de agrupamento, a tarefa envolve medir a precisão desses agrupamentos com métricas de aprendizado de máquina, como matriz de confusão e acurácia, o que adiciona um nível significativo de complexidade e avaliação ao projeto.

Flor Íris

O gênero íris é composto de cerca de 300 espécies de flores vistosas de várias cores, incluindo branco, amarelo, violeta, rosa e azul. A íris é o maior grupo da família Iridaceae no hemisfério norte, mas essas flores estão presentes em muitas áreas do mundo.

Ademais, é um gênero de flor que apresenta três espécies distintas: virgínica, setosa e versicolor, as quais podem ser caracterizadas por meio dos comprimentos de pétalas e sépalas (partes que sustentam o botão floral). Partindo dessa observação, foi construída uma base de dados contendo características morfológicas da flor Iris.

Íris Virgínica

Íris Vercicolor

Íris Setosa

Base de dados Flor Íris

O conjunto de dados flor Iris ou conjunto de dados Iris de Fisher é um conjunto de dados multivariados introduzido pelo estatístico e biólogo britânico Ronald Fisher em seu artigo de 1936, O uso de múltiplas medições em problemas taxonômicos, como um exemplo de análise discriminante linear. Às vezes, é chamado de conjunto de dados da íris de Anderson

porque Edgar Anderson coletou os dados para quantificar a variação morfológica das flores da íris de três espécies relacionadas. Duas das três espécies foram coletadas na Península de Gaspé, "todas do mesmo campo, colhidas no mesmo dia e medidas ao mesmo tempo pela mesma pessoa com a mesma aparelho".

Com base no modelo discriminante linear de Fisher, esse conjunto de dados se tornou um caso de teste típico para muitas técnicas de classificação estatística em aprendizado de máquina, como máquinas de vetores de suporte.

	A	B	C	D	E
1	Variety	Sepal Length	Sepal Width	Petal Length	Petal Width
2	Setosa	5.1	3.5	1.4	0.2
3	...				
4	Versicolor	6.4	3.2	4.5	1.5
5	...				
6	Virginica	5.8	2.7	5.1	1.9
7					

O conjunto de dados contém um conjunto de 150 registros com cinco atributos - comprimento da sépala, largura da sépala, comprimento da pétala, largura da pétala e espécies, sendo elas Setosas, Versicolors e Virginicas.

Implementação do Grafo com Base na Estrutura da Base de Dados

Na Tarefa 1_A, o objetivo principal é construir um grafo onde cada observação da base Iris representa um vértice. Com cada amostra contendo 4 informações sobre sua composição, podemos considerar cada uma como um ponto no espaço quadridimensional: (x, y, z, w), onde x é o valor Sepal Length, y é Sepal Width, z é o Petal Length e por fim w é o Petal Width.
Com isso, podemos criar uma matriz de adjacências com base nas distâncias euclidianas normalizadas (DEN) entre pares de vértices. Esses vértices estarão conectados quando a distância entre eles for igual ou inferior a 0,3, indicando similaridade. A complexidade envolve não só a criação da matriz de adjacência, mas também o cálculo e normalização das distâncias para definir as conexões.

Cálculo das distâncias euclidianas(DE):

```
for (int j = i + 1; j < numAmostras; j++)
{
    float dist = pow(dataset[i].sepal_length - dataset[j].sepal_length, 2) +
                 pow(dataset[i].sepal_width - dataset[j].sepal_width, 2) +
                 pow(dataset[i].petal_length - dataset[j].petal_length, 2) +
                 pow(dataset[i].petal_width - dataset[j].petal_width, 2);
    dist = sqrt(dist);
```

Normalização das distâncias (DEN):

```
85      for (int i = 0; i < numAmostras; i++)
86      {
87          for (int j = i + 1; j < numAmostras; j++)
88          {
89              float calc = (matrizAdjacencia[i][j] - minDist) / (maxDist - minDist);
90              if(calc > maxDistDEN){
91                  maxDistDEN = calc;
92                  amostraMaxDEN1 = i;
93                  amostraMaxDEN2 = j;
94              }
95              if(calc < minDistDEN){
96                  minDistDEN = calc;
97                  amostraMinDEN1 = i;
98                  amostraMinDEN2 = j;
99              }
100
101             matrizAdjacencia[i][j] = matrizAdjacencia[j][i] = calc < LIMIAR ? 1 : 0;
102
103             if (matrizAdjacencia[i][j] == 1)
104             {
105                 fprintf(arestas, "%i %i\n", i, j);
106             }
107         }
108         matrizAdjacencia[i][i] = 0;
109     }
```

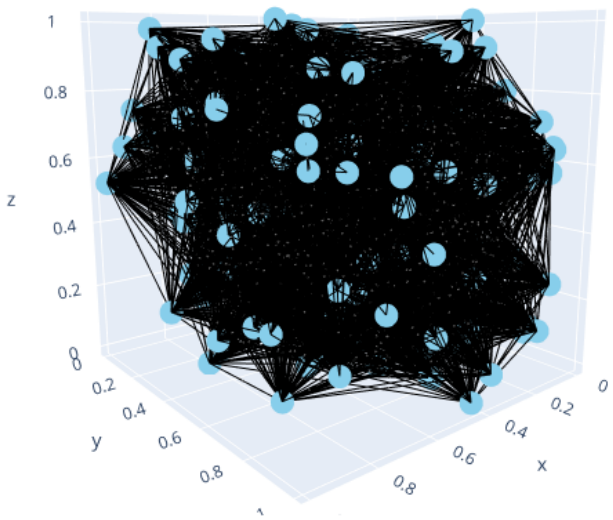
Na linha 105 do código acima é onde salvamos os vértices do grafo em um documento para posteriormente plotarmos esse grafo utilizando um algoritmo python. Além do mais, como requisito funcional da tarefa A, temos:

```
[..da Computacao/TEG - Teoria dos Grafos/Trabalhos/Grafo Íris/src]
base herton  master - ./output

O total de vértices lidos = 150
A maior DE e o par de vértices (vi vj) que a determinaram = 7.085196 (13, 118)
A menor DE e o par de vértices (vi vj) que a determinaram = 0.000000 (101, 142)
A maior DEN e o par de vértices (vi vj) que a determinaram = 1.000000 (13, 118)
A menor DEN e o par de vértices (vi vj) que a determinaram = 0.000000 (101, 142);

Matriz de Confusão:
```

Ao final do processo, teremos 150 vértices com cerca de 5250 arestas, sendo o plot em 3D:



Análise de Componentes Conexos e Clustering

Na Tarefa 1_B, é solicitado que o grafo construído seja utilizado para identificar componentes conexos, aplicando algoritmos de busca em largura (BFS) ou profundidade (DFS). O intuito é dividir o grafo em dois grandes agrupamentos, sendo que o maior grupo de setosa e o outro para as demais espécies. A tarefa destaca a necessidade de avaliar a eficácia desses agrupamentos, utilizando métricas de Machine Learning. Com isso, além da programação, há a necessidade de entender como adaptar algoritmos para detecção de componentes conexos e análise de clustering.

Componentes Conexos e Algoritmo BFS

O algoritmo de busca escolhido para verificar os componentes conexos foi o BFS, que consiste em começar pelo vértice raiz e explorar todos os vértices vizinhos. Então, para cada um desses vértices mais próximos, exploramos os seus vértices vizinhos inexplorados e assim por diante, até que ele encontre o alvo da busca ou visite todos os vértices.

```
142
143  // Função para contar componentes conexos e seus tamanhos
144  void contarComponentesConexos(float **matrizAdjacencia, int numAmostras)
145  {
146      bool *visitados = (bool *)calloc(numAmostras, sizeof(bool));
147      int numComponentes = 0;
148
149      printf("Componentes conexos:\n");
150
151      for (int i = 0; i < numAmostras; i++)
152      {
153          if (!visitados[i])
154          {
155              int tamanhoComponente = 0;
156              bfs(i, matrizAdjacencia, numAmostras, visitados, &tamanhoComponente);
157              printf("Componente %d: Tamanho %d\n", ++numComponentes, tamanhoComponente);
158          }
159      }
160
161      free(visitados);
162  }
163
```

Dessa forma, é possível identificar os componentes conexos e seus respectivos tamanhos.

Algoritmo K-Mens e clustering

O algoritmo escolhido foi o K-Mens, que é um algoritmo de clusterização (agrupamento) não supervisionado, baseado na definição de centroides que representam clusters. O “K” refere-se ao número de centroides (clusters) definidos previamente e o “Means” à média dos pontos em cada cluster que determina a posição de seu centroide.

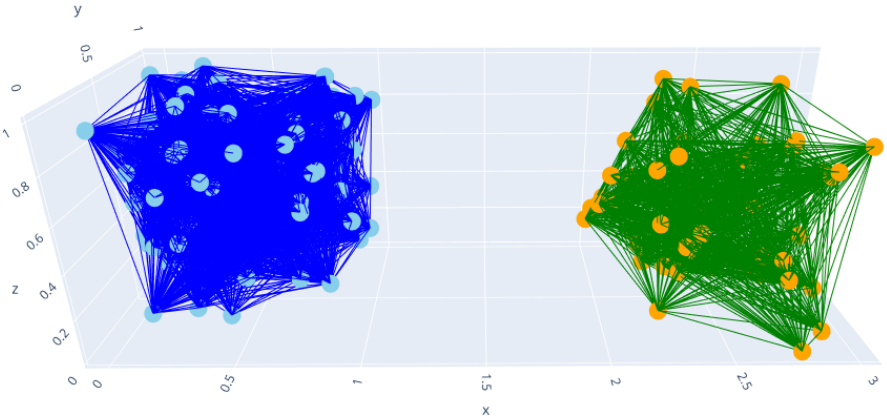
Primeiramente, vamos inicializar os centros dos clusters de maneira aleatória. Selecionando amostras aleatórias da base de dados dataset para serem os centros iniciais de cada cluster. O objetivo é garantir que cada centro escolhido seja único para evitar duplicações, verificando que o índice selecionado não foi previamente escolhido.

```
165
166 void inicializarCentros(Flor dataset[], Flor centros[], int numAmostras)
167 {
168     srand(time(NULL));
169     for (int i = 0; i < NUM_CLUSTERS; i++)
170     {
171         int indice;
172         bool unico;
173         do
174         {
175             indice = rand() % numAmostras;
176             unico = true;
177             for (int j = 0; j < i; j++)
178             {
179                 if (indice == j)
180                 {
181                     unico = false;
182                     break;
183                 }
184             }
185         } while (!unico);
186         centros[i] = dataset[indice];
187     }
188 }
189
```

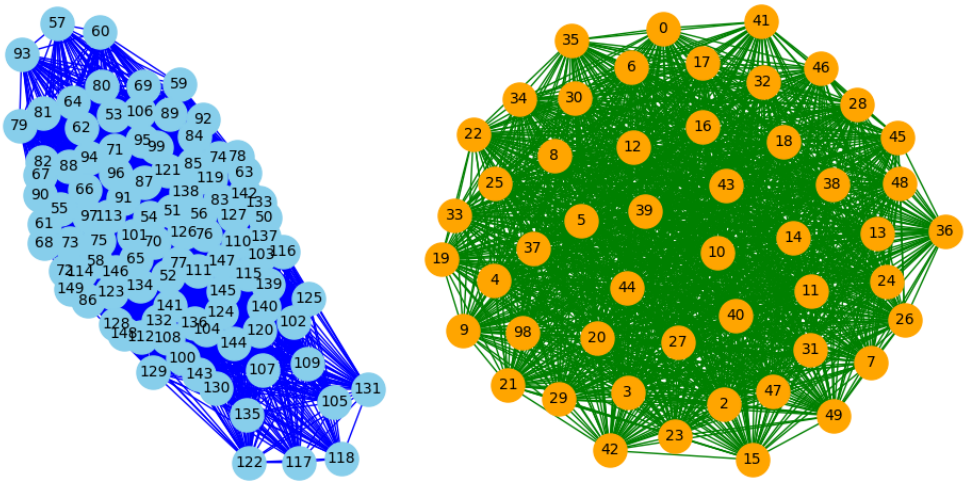
Agora, o seguinte processo se repete para ajustar os clusters até que os centros se estabilizem: atribuir cada amostra do conjunto de dados ao cluster mais próximo, baseado na distância euclidiana entre a amostra e o centro do cluster. Após atribuir cada amostra a um cluster, é recalculado os centros dos clusters com base nas médias das coordenadas das amostras pertencentes a cada cluster.

```
247 void kMeans(Flor dataset[], int numAmostras, int *clusters, float **matrizAdjacencia)
248 {
249     Flor centros[NUM_CLUSTERS];
250     inicializarCentros(dataset, centros, numAmostras);
251
252     for (int iter = 0; iter < MAX_ITER; iter++)
253     {
254         atribuirPontos(dataset, numAmostras, centros, clusters, matrizAdjacencia);
255         recalcularCentros(dataset, numAmostras, centros, clusters);
256     }
257 }
258
```

Ao final teremos os dois clusters salvos para o plote em 3D:



Em 2D:



Os gráficos revelam que o conjunto de Iris setosa (representado por vértices amarelos) possui uma estruturação mais clara em relação às demais espécies (Iris versicolor e Iris virginica, representadas por vértices azul-claro), uma vez que suas características morfológicas são mais distintas. Em particular, as variáveis de comprimento e largura das pétalas das setosas diferem significativamente das das outras espécies, o que facilita a formação de um cluster bem separado. Em contraste, versicolor e virginica tendem a ter medidas mais sobrepostas nessas variáveis, tornando a separação entre esses dois grupos mais desafiadora.

Avaliação de Performance e Métricas de Qualidade

Na Tarefa 1_C, o foco está na avaliação da qualidade dos agrupamentos obtidos. Essa avaliação é feita por meio da construção de uma matriz de confusão e do cálculo da acurácia, comparando os agrupamentos obtidos pelo grafo com as classes reais da base de dados Iris. A matriz de confusão permite identificar verdadeiros positivos (TP), falsos positivos (FP), verdadeiros negativos (TN) e falsos negativos (FN), auxiliando na extração de métricas de desempenho como precisão e recall. A análise desses valores oferece uma compreensão mais profunda da precisão do modelo de clustering implementado.

```
311  void acuracia(Flor dataset[], int numAmostras, int *clusters) {
312      int TP = 0, FP = 0, TN = 0, FN = 0;
313
314      for (int i = 0; i < numAmostras; i++) {
315          // Caso a flor seja "Setosa" e esteja no cluster 1 -> Verdadeiro Positivo
316          if (strcmp(dataset[i].especie, "Setosa") == 0 && clusters[i] == 1) {
317              TP++;
318          }
319          // Caso a flor seja "Setosa" mas esteja no cluster 0 -> Falso Negativo
320          else if (strcmp(dataset[i].especie, "Setosa") == 0 && clusters[i] == 0) {
321              FN++;
322          }
323          // Caso a flor não seja "Setosa" e esteja no cluster 0 -> Verdadeiro Negativo
324          else if (strcmp(dataset[i].especie, "Setosa") != 0 && clusters[i] == 0) {
325              TN++;
326          }
327          // Caso a flor não seja "Setosa" mas esteja no cluster 1 -> Falso Positivo
328          else if (strcmp(dataset[i].especie, "Setosa") != 0 && clusters[i] == 1) {
329              FP++;
330          }
331      }
332
```

Portanto, chegamos na seguinte matriz de confusão:

```
Matriz de Confusão:
                 Predito Setosa    Predito Nao Setosa
Real Setosa:        48 (TP)          1 (FN)
Real Nao Setosa:    2 (FP)          99 (TN)
Acuracia: 98.00%
```

Ao final, obtivemos 98% de acurácia com o modelo.

Considerações Finais

Essas tarefas cobrem desde a construção básica de estruturas de dados complexas, como grafos, até a aplicação de conceitos de Machine Learning para avaliar e otimizar a eficácia do modelo de agrupamento. As etapas exigem habilidades tanto de implementação de algoritmos como de análise crítica de desempenho, proporcionando uma abordagem prática e aprofundada de técnicas de classificação e agrupamento em aprendizado de máquina

Referências

- https://www.primaveragarden.com.br/iris-a-flor-de-lis/
- https://pt.wikipedia.org/wiki/Conjunto_de_dados_flor_Iris
- https://docs.ufpr.br/~volmir/PO_II/A_5_A_grafos.pdf
- https://github.com/mayursrt/k-means-on-iris-dataset/blob/main/Prediction%20using%20Unsupervised%20ML.ipynb
- https://lamfo-unb.github.io/2017/10/05/Introducao_basica_a_clusterizacao/