Atividade 4 **Yan A. S. Duarte**
PEL 208 Tópicos Especiais em Aprendizagem
Prof. Reinaldo A. C. Bianchi Entrega: 01/11/2017

Introdução

Esse relatório tem como objetivo detalhar a teoria, a implementação, os resultados e a conclusão da quarta atividade do curso. A propósta do exercício é implementar o método k-Means.

Teoria

k-Means

k-Means clustering é um método de quantização vetorial que tem como objetivo separar amostras, de acordo com suas características, em um determinado número de clusters. Para alcançar esse objetivo, o algoritmo inicia com K pontos centrais de clusters e executa dois passos. O primeiro consiste em separas um conjunto de amostras que estejam mais próximas a um ponto central, formando assim os clusters. O segundo ponto calcula a média das amostras que estão em cada cluster para formar um novo ponto central. O processo é repetido até que o problema convirja.

Matematicamente, para um conjunto k de centros $m_1^{(1)}, ..., m_k^{(1)}$ a formula para a etapa de atribuição se da por:

$$S_i^{(t)} = \{x_p : \|x_p - m_i^{(t)}\|^2 \leq \|x_p - m_j^{(t)}\|^2 \forall j, 1 \leq j \leq k\}$$

onde x_p é atribuido somente para um $S^{(t)}$, mesmo que pudesse ser atribuido para dois ou mais.

Já a etapa de atualização, as médias são calculadas a partir da fórmula:

$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

Implementação

Para a elaboração do exercício foi utilizada a linguagem de programação C++. Foram criadas três classes diferentes, uma para o controle de cada ponto, uma para o controle do cluster e uma geral para o kmeans.

classe Point

Essa classe foi criada para controlar os pontos e a qual cluster eles pertencem. Suas funções são:

```
class Point{
    private:
        int id_point, id_cluster;
        vector<double> values;
        int total_values;
        string name;
    public:
        Point(int id_point, vector<double>& values, string name = "");
```

```
            int get_id();
            void set_cluster(int id_cluster);
            int get_cluster();
            double get_value(int index);
            int get_total_values();
            void add_value(double value);
            string get_name();
    };
```

Classe Cluster

Essa classe foi criada para adicionare remover pontos e também controlar os clusters. Suas funções são:

```
    class Cluster{
        private:
            int id_cluster;
            vector<double> central_values;
            vector<Point> points;
        public:
            Cluster(int id_cluster, Point point);
            void add_point(Point point);
            bool remove_point(int id_point);
            double get_central_value(int index);
            void set_central_value(int index, double value);
            Point get_point(int index);
            int get_total_points();
            int get_id();
    };
```

classe kmeans

A classe Kmeans foi desenvolvida para executar o algoritmo kmeans, passando o número de iterações e os valores dos pontos que serão analisados.

```
    class KMeans{
        private:
            int K;
            int total_values, total_points, max_iterations;
            vector<Cluster> clusters;
            int get_id_nearest_center(Point point);
        public:
            KMeans(int K, int total_points, int total_values, int max_iterations);
            void run(vector<Point> & points);
    };
```

Os códigos completos podem ser vistos no arquivo matrix.h que se encontra na raiz da pasta principal.

Testes

Para testar o funcionamento do kmeans foram utilizados os datasets:

- Exemplo do slide;
- Iris dataset: banco com 150 amostras de plantas divididas iguamente entre 3 classes: iris setosa, iris versicolour e iris virginica. Os atributos são: comprimento da sepala em cm, largura da sepala em cm, comprimento da petala em cm e largura da petala em cm.

- Seeds dataset: banco com 210 amostras de sementes de plantas divididas igualmente entre 3 classes: canadian, kama e rosa. Os atributos são: area, perimeter, compactness, largura do núcleo, comprimento do núcleo, coeficiente de assimetria e comprimento do sulco do núcleo;
- Lenses dataset: banco com 24 amostras de pacientes que necessitam utilizar lentes de contato divididas entre 3 classes: paciente necessita lentes de contato duras, paciente necessita lentes de contato macias e paciente não necessita de lentes de contato. Os atributos são: idade do paciente, prescrição de espetáculo, astigmática e taxa de produção de lágrimas.

Resultados

Exemplo do slide

Cluster 1

Valor do cluster: $[2,62857 \quad 6,5]$
total de pontos: 7

Cluster 2

Valor do cluster: $[1,36667 \quad 2,15]$
total de pontos: 6

Cluster 3

Valor do cluster: $[4,775 \quad 3,05]$
total de pontos: 4

Como o exemplo do slide é bidimensional, conseguimos representa-lo em um gráfico:

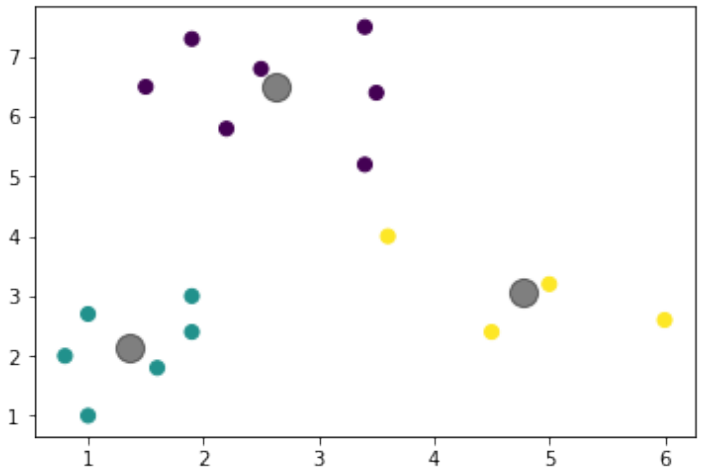

Figura 1: Resultado com dataset do exemplo do slide.

Iris Dataset

Cluster 1

Valor do cluster: $\begin{bmatrix} 5,88 & 2,74 & 4,39 & 1,43 \end{bmatrix}$
iris-versicolor: 47
iris-virginica: 14
iris-setosa: 0
total de pontos: 61

Cluster 2

Valor do cluster: $\begin{bmatrix} 6,86 & 3,08 & 5,72 & 2,05 \end{bmatrix}$
iris-versicolor: 3
iris-virginica: 36
iris-setosa: 0
total de pontos: 39

Cluster 3

Valor do cluster: $\begin{bmatrix} 5,01 & 3,42 & 1,46 & 0,24 \end{bmatrix}$
iris-versicolor: 0
iris-virginica: 0
iris-setosa: 50
total de pontos: 50

Seeds Dataset

Cluster 1

Valor do cluster: $\begin{bmatrix} 11,9644 & 13,2748 & 0,8522 & 5,22929 & 2,87292 & 4,75974 & 5,08852 \end{bmatrix}$
Canadian: 68
Kama: 9
Rosa: 0
total de pontos: 77

Cluster 2

Valor do cluster: $\begin{bmatrix} 14,6485 & 14,4604 & 0,879167 & 5,56378 & 3,2779 & 2,64893 & 5,19232 \end{bmatrix}$
Canadian: 2
Kama: 60
Rosa: 10
total de pontos: 72

Cluster 3

Valor do cluster: $\begin{bmatrix} 18,7218 & 16,2974 & 0,885087 & 6,20893 & 3,72267 & 3,60359 & 6,0661 \end{bmatrix}$
Canadian: 0

Kama: 1
Rosa: 60
total de pontos: 61

Lenses Dataset

Cluster 1

Valor do cluster: $\begin{bmatrix} 1,46154 & 1,46154 & 1,07692 & 3 \end{bmatrix}$
hard-lenses: 4
soft-lenses: 4
no-lenses: 5
total de pontos: 13

Cluster 2

Valor do cluster: $\begin{bmatrix} 2 & 2 & 2 & 3 \end{bmatrix}$
hard-lenses: 0
soft-lenses: 1
no-lenses: 1
total de pontos: 2

Cluster 3

Valor do cluster: $\begin{bmatrix} 1,44444 & 1,44444 & 2 & 1,55556 \end{bmatrix}$
hard-lenses: 4
soft-lenses: 3
no-lenses: 2
total de pontos: 9

Conclusão

Nesse relatório foi implementado, em linguagem c++, um algoritmo para executar o método kmeans e, para testar o algoritimo, foram utilizados os datasets de exemplo da aula, iris, seeds e lenses.

Os resultados obtidos conseguiram separar as classe de cada dataset de uma maneira satisfatória.

Referências

[1] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, 2001.*

[2] k-means clustering Disponível em (https://en.wikipedia.org/wiki/K-means_clustering). Acesso em: 30 de out. de 2017

[3] *Iris dataset Disponível em (http://archive.ics.uci.edu/ml/datasets/iris). Acesso em: 30 de out. de 2017*

[4] Seeds dataset Disponível em (http://archive.ics.uci.edu/ml/datasets/seeds). Acesso em: 30 de out. de 2017

[5] *Lenses dataset Disponível em (http://archive.ics.uci.edu/ml/datasets/Lenses). Acesso em: 30 de out. de 2017*