

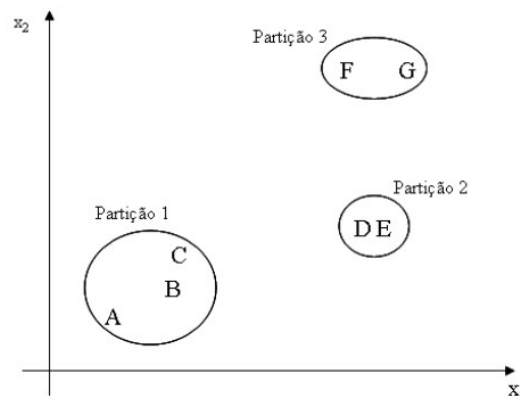
Primeiro Trabalho de Inteligência Artificial e Sistemas Inteligentes

Prof. Flávio Miguel Varejão

I. Descrição do Problema

Agrupamento de dados multidimensionais é um dos problemas mais comuns na área de aprendizado de máquina. Esse problema consiste em dividir um conjunto de pontos em um espaço multidimensional em um determinado número pré-especificado de grupos de modo que os pontos pertencentes a um mesmo grupo estão mais relacionados entre si e menos relacionados em relação aos pontos associados aos outros grupos.

A figura abaixo ilustra um exemplo de agrupamento no qual os sete pontos {A, B, C, D, E, F, G} foram agrupados em três grupos, indicando que os padrões {A, B, C} são mais similares entre si do que em relação aos demais, assim como os padrões {D, E} e {F, G}.



Formalmente, dado um conjunto de dados X com N pontos $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, sendo que cada ponto $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{id}]^T$ possui d coordenadas (dimensões), deseja-se encontrar K grupos $\{C_1, \dots, C_K\}$, de tal forma que as seguintes condições sejam atendidas:

- $C_j \neq \emptyset, j = 1, \dots, K$
- $\bigcup_{j=1}^K C_j = X$
- $C_i \cap C_j = \emptyset, i \neq j, i, j = 1, \dots, K$

Além de atender essas condições, a qualidade da divisão em grupos deve ser avaliada. Neste trabalho o critério de qualidade utilizado será a soma das distâncias euclidianas quadradas (SSE) entre os pontos pertencentes a cada um dos grupos:

$$SSE = \sum_{j=1}^K \sum_{\mathbf{x}_i \in C_j} \|\mathbf{x}_i - \mu_j\|^2$$

onde $\|\mathbf{x}_i - \mu_j\|$ é a distância Euclidiana entre o ponto \mathbf{x}_i e o centróide μ_j .

O centróide $\mu_j = [\mu_{j1}, \mu_{j2}, \dots, \mu_{jd}]^T$ é o ponto representativo do grupo C_j e é calculado como o centro de massa do grupo:

$$\mu_j = \frac{1}{n_j} \sum_{x_i \in C_j} x_i$$

onde n_j é o total de pontos pertencentes ao grupo C_j .

A distância Euclideana $\|x_i - \mu_j\|$ é calculada pela expressão:

$$\|x_i - \mu_j\| = \sqrt{(x_{i1} - \mu_{j1})^2 + (x_{i2} - \mu_{j2})^2 + \dots + (x_{id} - \mu_{jd})^2}$$

Neste problema busca-se a distribuição de pontos em grupos que minimiza a SSE.

Este trabalho consiste em realizar uma comparação experimental entre um conjunto pré-definido de metaheurísticas aplicadas ao problema de agrupamento de dados. As metaheurísticas escolhidas são: Simulated Annealing, Genetic Algorithm e GRASP. O procedimento experimental será dividido em duas etapas: a primeira etapa consiste no ajuste de hiperparâmetros das metaheurísticas e a segunda etapa consiste na comparação das metaheurísticas considerando apenas os valores dos hiperparâmetros selecionados na primeira etapa. Os problemas utilizados para a etapa de treino são necessariamente distintos dos problemas utilizados na etapa de teste. Para limitar o tempo de execução total dos experimentos, cada execução para a resolução de um problema deve ser limitada a um tempo máximo.

1. Treino

Na primeira etapa da experimentação será necessário realizar uma busca em grade (grid search) considerando os seguintes valores de hiperparâmetros de cada metaheurística:

Simulated Annealing: {To: [500, 100, 50]; alfa: [0.95, 0.85, 0.7]; numIter (número de iterações): [350, 500]}

GRASP: {numIter (número de iterações): [20, 50, 100, 200, 350, 500], numBest (numero de melhores elementos): [5, 10, 15]}

Genetic Algorithm: {Tamanho da população: [10, 30, 50], taxa de crossover: [0.75, 0.85, 0.95], taxa de mutação: [0.10, 0.20]}

A condição de parada das metaheurísticas deve ser uma combinação entre algum critério definido na própria metaheurística e o tempo máximo de execução de 1 segundo.

O pseudo-código seguinte mostra os passos a serem realizados para o ajuste de hiperparâmetros das metaheurísticas:

Para cada método faça

Para cada problema de treino faça

Para cada configuração de hiperparâmetros faça

Executar 10 vezes e obter média e tempo médio

fim-se

Padronizar resultados da média com z-score

Fazer ranqueamento

```
fim-se
Obter média, desvio padrão e ranqueamento médio da configuração
Obter melhor configuração por média e por ranqueamento médio do método
Obter as 5 melhores resultados de médias padronizadas e os
tempos correspondentes das configurações de cada método
Obter ranqueamento obtido por cada configuração de
método em cada problema e seu ranqueamento médio
fim-se
Retornar tabela com melhor configuração de cada método por média e
ranqueamento médio
```

2. *Teste*

A segunda etapa da experimentação consiste em avaliar as metaheurísticas no conjunto de problemas de teste com os valores de hiperparâmetros selecionados na etapa de treino.

O pseudo-código seguinte mostra os passos a serem realizados para avaliação e comparação das metaheurísticas:

```
Para cada problema de teste faça
    Para cada método com melhor configuração faça
        Executar 20 vezes e obter média e tempo médio de execução
    fim-se
    Executar o método k-means 20 vezes e obter média e tempo médio de
    execução
    Padronizar resultados da média com z-score
    Fazer ranqueamento
fim-se
Obter média padronizada e desvio padrão, tempos correspondentes e ranqueamento
médio de cada método
Realizar testes estatísticos para comparar os métodos par a par
    Usar teste t pareado e wilcoxon
Obter tabela pareada com resultados
Retornar melhor método em geral por média e por ranqueamento médio
```

3. *Artigo*

Após a realização dos experimentos, um artigo descrevendo todo o processo experimental realizado deverá ser escrito em latex usando o software overleaf. O artigo deve ser estruturado contendo os seguintes componentes:

1. Título

2. Resumo
3. Seção 1. Introdução
4. Seção 2. Descrição do Problema de K Médias
5. Seção 3. Descrição dos Métodos Utilizados
 - a. Seção 3.1 Representação do Descritor de Espaço de Estados
 - b. Seção 3.2 GRASP
 - c. Seção 3.3 Simulated Annealing
 - d. Seção 3.4 Genetic Algorithm
6. Seção 4. Descrição dos Resultados dos Experimentos
 - a. Seção 4.1 Treino
 - i. Apresentação de tabela com os valores dos hiperparâmetros das cinco melhores configuração de cada método
 - ii. Apresentação dos bloxplots de média e de tempo de cada método
 - iii. Apresentação de tabela com ranqueamento em cada problema e ranqueamento médio de cada método
 - iv. Apresentação de tabela com melhor configuração de cada método por média e por ranqueamento médio
 - v. Análise dos resultados alcançados
 - b. Seção 4.2 Teste
 - i. Apresentação da tabela contendo média padronizada, desvio padrão, média e desvio padrão dos tempos de execução de todas os métodos testados
 - ii. Apresentação dos bloxplots de média e de tempo de cada método
 - iii. Apresentação de tabela contendo os ranqueamentos das métodos para cada problema de teste e o ranqueamento médio
 - iv. Apresentar tabela pareada de testes estatísticos com destaque nos pares de métodos onde houve diferença significativas
 - v. Apresentar melhor método geral por média e ranqueamento médio
 - vi. Análise dos resultados alcançados
7. Seção 5. Conclusões
 - a. Análise geral dos resultados
 - b. Contribuições do Trabalho
 - c. Melhorias e trabalhos futuros
8. Referências Bibliográficas

4. Informações Complementares

Os problemas a serem usados para treino e teste estão descritos nos apêndices A e B deste enunciado.

Os gráficos bloxplot requeridos no treino e no teste devem ser gerados usando função específica do pacote seaborn (ver instruções de instalação e uso no apêndice C deste enunciado).

O apêndice D deste enunciado apresenta instruções de instalação e uso do overleaf para a escrita do artigo.

O apêndice E contém instruções de como obter as bases de dados utilizadas no trabalho.

5. Condições de Entrega

O trabalho deve ser feito individualmente e submetido pelo sistema da sala virtual até a data limite (24 de março de 2021).

O trabalho deve ser submetido em dois arquivos: um arquivo pdf com o artigo produzido no trabalho e todos os arquivos com código fonte em python utilizados em um arquivo zip. Tanto o arquivo pdf quanto o zip devem possuir o mesmo nome Trab1_Nome_Sobrenome. Note que a data limite já leva em conta um dia adicional de tolerância para o caso de problemas de submissão via rede. Isso significa que o aluno deve submeter seu trabalho até no máximo um dia antes da data limite. Se o aluno resolver submeter o trabalho na data limite, estará fazendo isso assumindo o risco do trabalho ser cadastrado no sistema após o prazo. Em caso de recebimento do trabalho após a data limite, o trabalho não será avaliado e a nota será ZERO. Aluno que receber zero por este motivo e vier pedir para o professor considerar o trabalho não será considerado. Plágio ou cópia de trabalhos serão verificadas automaticamente por sistemas como o moss. Trabalhos em que se configure cópia receberão nota zero independente de quem fez ou quem copiou.

6. Requisitos da implementação

- Modularize seu código adequadamente.
- Crie códigos claros e organizados. Utilize um estilo de programação consistente, Comente seu código.
- Os arquivos do programa devem ser lidos e gerados na mesma pasta onde se encontram os arquivos fonte do seu programa.

Observação importante

Caso haja algum erro neste documento, serão publicadas novas versões e divulgadas erratas em sala de aula. É responsabilidade do aluno manter-se informado, freqüentando as aulas ou acompanhando as novidades na página da disciplina na sala virtual.

Apêndice A. Problemas de Treino

Base de dados Iris - $k = [3, 7, 10, 13, 22]$

Base de dados Wine - $k = [2, 6, 9, 11, 33]$

Apêndice B. Problemas de Teste

Base de dados Iris - $k = [2, 4, 8, 11, 15, 17, 23, 28, 32, 50]$

Base de dados Wine - $k = [3, 5, 13, 15, 20, 23, 25, 30, 41, 45]$

Base de dados Ionosphere - $k = [2, 3, 5, 10, 15, 20, 25, 30, 40, 50]$

Apêndice C. Boxplots usando seaborn

Instruções de instalação para python3:

```
sudo apt install python3 python3-pip
pip3 install seaborn
```

```
# arquivo bloxplot.py
# library & dataset
import seaborn as sns
Instruções de instalaçãoimport matplotlib.pyplot as plt
```

```
def example1():
    mydata=[1,2,3,4,5,6,12]
    sns.boxplot(y=mydata) # Also accepts numpy arrays
    plt.show()

def example2():
    df = sns.load_dataset('iris')
    #returns a DataFrame object. This dataset has 150 examples.
    #print(df)
    # Make boxplot for each group
    sns.boxplot( data=df.loc[:,:] )
    # loc[:,:] means all lines and all columns
    plt.show()

example1()
example2()
```

Uso em python3:

```
python3 boxplot.py
```

Apêndice D. Artigo em Latex usando Overleaf

Juntamente com este enunciado foi disponibilizado um arquivo zip com o template de latex para confecção do artigo. O primeiro passo a ser feito é criar uma conta pessoal no Overleaf (<https://www.overleaf.com/register>). Uma vez criada sua conta, deve-se entrar nela. Para incluir o template no overleaf, basta apenas selecionar "New Project>Upload Project" e selecionar o arquivo zip, como mostrado na figura abaixo. Não é necessário descompactar, faça o upload do zip direto. Lembrar de renomear o artigo após o upload do arquivo.

<input type="checkbox"/> Title	Owner
<input type="checkbox"/> On the analysis of CLR ● Artigos x	You
<input type="checkbox"/> Simulation Multi-label Distribution 2019 - TKDE ● Artigos x	You
<input type="checkbox"/> Simulation Multi-label Distribution (IS-2019) ● Artigos x	You
<input type="checkbox"/> Simulation Multi-label Distribution (Information and Management) ● Artigos x	You
<input type="checkbox"/> Simulation Multi-label Distribution (Data mining and Knowledge) ● Artigos x	You
<input type="checkbox"/> Coverage_NPHard_PRletters-Revised-Marked ● Artigos x	You
<input type="checkbox"/> Simulation Multi-label Distribution ● Artigos x	You
<input type="checkbox"/> Coverage_NPHard_PRletters (Revised) (final) ● Artigos x	You
<input type="checkbox"/> Coverage_NPHard_PRletters ● Artigos x	You
<input type="checkbox"/> Coverage_NPHard_jmlr ● Artigos x	You
<input type="checkbox"/> contribution ● Artigos x	You

You are using the free

Apêndice E. Bases de dados

```
import numpy as np
from sklearn.datasets import load_iris, load_wine
from pandas import read_csv
```

```
iris = load_iris()
wine = load_wine()
ionos = read_csv('~/.ionosphere.data') # colocar path apropriada
ionosphere = np.asarray([i[:34] for i in ionosphere])
```

ionosphere.data pode ser obtido em <http://archive.ics.uci.edu/ml/datasets/Ionosphere>