15/04/2014

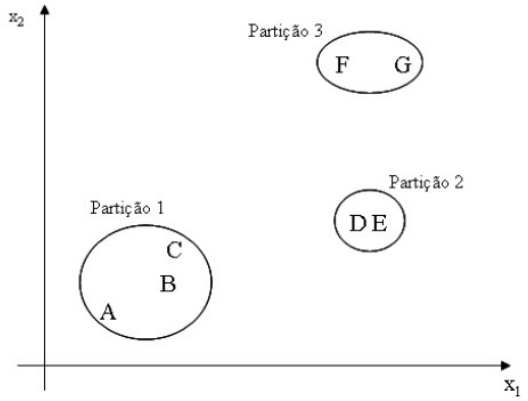
Primeiro Trabalho de LP

Prof. Flávio Miguel Varejão

I. Descrição do Problema

Agrupamento de dados multidimensionais é um dos problemas mais comuns na área de mineração de dados. 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 $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, sendo que cada ponto $\mathbf{x_i} = [x_{i1}, x_{i2}, \ldots, x_{id}]_t$ possui d coordenadas (dimensões), deseja-se encontrar K grupos $\{C_1, \ldots, C_K\}$, de tal forma que as seguintes condições sejam atendidas:

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

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:

$$SSE = \sum_{j=1}^{K} \sum_{\mathbf{x_i} \in C_j} \|\mathbf{x_i} - \mu_\mathbf{j}\|^2$$

onde $\| x_i - \mu_j \|$ é a distância Euclidiana entre o ponto x_i e o centróide μ_j.

O centróide $\mu_\mathbf{j} = [\mu_{j1}, \mu_{j2}, \ldots, \mu_{jd}]_t$ é o ponto representativo do grupo C_j e é calculado

como o centro de massa do grupo:

$$\mu_{\mathbf{j}} = \frac{1}{n_j} \sum_{\mathbf{x_i} \in C_j} \mathbf{x_i}$$

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

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

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

Neste trabalho será necessário implementar o algoritmo de agrupamento descrito a seguir:

1.Selecionar K pontos iniciais como centróides dos grupos

   a.Selecionar o ponto cujas coordenadas somadas tem valor mínimo (em caso de empate, selecionar aquela cujas primeiras coordenadas tem menor valor)

   b.Selecionar o ponto mais distante do ponto inicial (em caso de empate, selecionar aquele cujas primeiras coordenadas tem menor valor)

   c.Selecionar sucessivamente, até completar os K pontos iniciais, o ponto mais distante do centróide do grupo formado pelos pontos selecionados até então (em caso de empate, selecionar aquele cujas primeiras coordenadas tem menor valor)

2.ENQUANTO Grupos se alteram em duas rodadas consecutivas ou não se completou N iterações FAÇA

   a.Distribuir os pontos em K grupos de acordo com a distância mais próxima do ponto para os centróides dos grupos (em caso de empate, atribuir o ponto ao grupo do centróide cujas primeiras coordenadas tem menor valor)

   b.Recalcular os centróides da nova distribuição de pontos em K grupos

3.Retornar a SSE da divisão em grupos e os grupos formados

II. Especificação do Sistema

Funcionalidades a serem implementadas:

1.Leitura do número de grupos a serem formados passado como parâmetro de entrada do programa. Por exemplo, para a formação de 3 grupos a chamada do programa será *python trab1.py 3*
2.Leitura dos dados dos pontos de um arquivo texto denominado "entrada.txt". Cada linha corresponde a um ponto. As coordenadas de um ponto são colocadas sucessivamente em uma linha separadas por espaço
3. Executar o algoritmo de agrupamento
4.Gravação do valor da SSE do resultado do agrupamento em um arquivo denominado"result.txt". O valor da SSE deve ter obrigatoriamente 4 casas decimais. Use os comandos *print("%.4f" % SSE)* ou *write("%.4f" % SSE)* para imprimir o valor da variável SSE e obter essa funcionalidade

5. Gravação dos pontos de cada grupo em um arquivo chamado “saida.txt”. Cada ponto é representado pelo número da linha no qual foi registrado no arquivo entrada.txt. Os números que representam os pontos de cada grupo devem ser colocados em uma ou mais linhas sequenciais separados por espaços. Os números de cada grupo devem ser apresentados em ordem crescente. A separação dos números de um grupo para outro será marcada por uma linha em branco

Formato dos Dados do Sistema:

```
número de grupos (k):    inteiro não negativo
coordenadas dos pontos:  inteiro ou ponto flutuante
SSE:                     ponto flutuante
número das linhas:       inteiro não negativo
```

Exemplo de arquivo entrada.txt:

```
7 5.4 6.32 9
17 32.3 5 9.99
33 54 5.6 65.8
77.7 33.4 98 7.56
8.9 5.8 6 9
```

Exemplo de arquivo result.txt:

```
988.3217
```

Exemplo de arquivo saida.txt:

```
2 6

3 5

4
```

III. Requisitos da implementação

1.Modularize seu código adequadamente. O uso de variáveis globais é proibido.

2.Crie códigos claros e organizados. Utilize um estilo de programação consistente, Comente seu código.

3.Os arquivos do programa devem ser lidos e gerados na mesma pasta onde se encontra os arquivos fonte do seu programa.

IV. Condições de Entrega

O trabalho deve ser feito individualmente e submetido por e-mail até as 23:59 horas de 11 de maio de 2014. Note que as datas limites já levam 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 estará cometendo um ato de DESRESPEITO ao professor e estará sujeito a perda adicional de pontos na média.

V. Formato de Entrega dos Trabalhos

O recebimento dos trabalhos é automatizado. Portanto, as regras a seguir devem ser seguidas à risca para evitar que seu trabalho não possa ser avaliado.

O código-fonte de sua solução deverá ser compactado e entregue por e-mail (anexo ao e-mail) para o endereço fvarejao@ninfa.inf.ufes.br.

Serão aceitos trabalhos entregues até as 23h59 da data limite. O assunto do e-mail deverá ser o seguinte:

```
lp:trab<id>:<nome>:
```

O termo "<id>" deve ser substituído pelo número correspondente do trabalho (1 ou 2). O termo "<nome>" deverá ser substituído pelo nome e o último sobrenome do aluno, sem acentos, til ou cedilha, como no exemplo abaixo:

```
lp:trab1:Flavio Varejao:
```

Atenção: o e-mail não deve ser enviado por servidores de emails que não seguem padrões normais de envio, tais como, TERRA, HOTMAIL ou BOL, pois o recebimento automatizado não consegue reconhecer seu trabalho.

O arquivo compactado deve estar no formato tar.gz com o nome trab<id>.tar.gz e conter apenas os arquivos fonte do programa (não deve conter executáveis ou arquivos compilados). Para isso, abra um console, mude o diretório de trabalho para a pasta onde se encontra o código-fonte do trabalho e execute o seguinte comando (no caso do trabalho 1):

```
tar -zcvf trab1.tar.gz *
```

Preste bastante atenção para fazer com que o codigo fonte não seja colocado em subdiretórios dentro do arquivo compactado. Se isso ocorrer a correção automática não funcionará e sua nota será ZERO. Atente também que os nomes usados no arquivo principal dos trabalhos DEVEM ser: trab1.py e trab2.py.

Um exemplo de um e-mail de envio do trabalho:

```
Para: fvarejao@ninfa.inf.ufes.br
De: Joao da Silva
Assunto: lp:trab1:Joao Silva:
Anexo: trab1.tar.gz
```

Para a execução dos programas, em princípio, será utilizada a versão 2 do python instalada no labgrad. Caso haja alguma modificação de versão de correção, ela será divulgada oportunamente. Os programas serão executados no sistema operacional linux. Para que não haja problemas na correção do seu trabalho e você seja prejudicado, garanta que ele é executado no sistema operacional linux na versão do python especificada.

Se tudo correr bem, você receberá um e-mail de confirmação do recebimento do trabalho. Neste e-mail haverá um hash MD5 do arquivo recebido. Para garantir que o arquivo foi recebido sem ser corrompido, gere o hash MD5 do arquivo que você enviou e compare com o hash recebido na confirmação. Para gerar o hash, utilize o seguinte comando:

```
md5sum trab1.tar.gz
```

Caso você não receba o e-mail de confirmação ou caso o valor do hash seja diferente, envie o trabalho novamente.

VI. Avaliação

Os trabalhos terão nota zero se:

- A data de entrega for fora do prazo estabelecido;
- O trabalho não gerar o arquivo com o resultado e formato esperado;
- For detectada a ocorrência de plágio pelo sistema.

Ainda, os trabalhos poderão ser avaliados segundo os seguintes critérios:

- Cumprimento das restrições estabelecidas no ítem III deste documento;
- Modularização (considerando o uso de arquivos separados para os diversos tipos abstratos de dados);
- Ausência de uso de variáveis globais;
- Legibilidade (nomes de variáveis bem escolhidos, código bem formatado, uso de comentários quando necessário, etc.);
- Consistência (utilização de um mesmo padrão de código);
- Eficiência (sem exageros, tentar evitar grandes desperdícios de recursos);

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 Internet.