



Universidade do Minho
Escola de Engenharia

Mineração de Dados

Mini Teste 3 - Individual
MiEI - 4º Ano - 1º Semestre

A85308 Filipe Miguel Teixeira Freitas Guimarães

Braga,
5 de janeiro de 2021

- 1 Considere o dataset “unbalanced”. Compare o desempenho dos algoritmos k-means e EM quando o número de partições é determinado pelo algoritmo EM. Considerar o número de partições (clusters) obtidas. Deve “desligar” o atributo classe para tornar mais real a avaliação. Compare estes resultados com os obtidos pelo algoritmo DBSCAN.

Executei primeiramente o algoritmos EM, ignorando a classe *Outcome* para tornar mais real a avaliação.

```

Class attribute: Outcome
Classes to Clusters:

  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 <-- assigned to cluster
  3  1  0  1  1  0  0  1  0  0  1  1  0  3  0  0  0  0 | Active
42 24 133 41 25 56 46 37 22 54 68 36 55 55 13 61 48 28 | Inactive

Cluster 0 <-- No class
Cluster 1 <-- No class
Cluster 2 <-- Inactive
Cluster 3 <-- No class
Cluster 4 <-- No class
Cluster 5 <-- No class
Cluster 6 <-- No class
Cluster 7 <-- No class
Cluster 8 <-- No class
Cluster 9 <-- No class
Cluster 10 <-- No class
Cluster 11 <-- No class
Cluster 12 <-- No class
Cluster 13 <-- Active
Cluster 14 <-- No class
Cluster 15 <-- No class
Cluster 16 <-- No class
Cluster 17 <-- No class

Incorrectly clustered instances :      720.0    84.1121 %

```

Consegue-se verificar que foram detectados 18 *clusters*, sendo o *Cluster 2* para a classe *Inactive*, o *Cluster 13* para a classe *Active* e os restantes não têm nenhuma classe associada. Verifica-se também que as instâncias incorretas de *clusters* são 84.1121%. Sendos os resultados por cluster os abaixo representados.

```
=== Model and evaluation on training set ===
```

```
Clustered Instances
```

0	45	(5%)
1	25	(3%)
2	133	(16%)
3	42	(5%)
4	26	(3%)
5	56	(7%)
6	46	(5%)
7	38	(4%)
8	22	(3%)
9	54	(6%)
10	69	(8%)
11	37	(4%)
12	55	(6%)
13	58	(7%)
14	13	(2%)
15	61	(7%)
16	48	(6%)
17	28	(3%)

Executei, de seguida, o algoritmo *KMeans*, ignorando, mais uma vez, a classe *Outcome*. Desta vez, e porque este algoritmo não prevê o número de clusters teve de se colocar nas definições.

```
=== Model and evaluation on training set ===
```

```
Clustered Instances
```

0	22	(3%)
1	19	(2%)
2	75	(9%)
3	27	(3%)
4	52	(6%)
5	122	(14%)
6	39	(5%)
7	102	(12%)
8	52	(6%)
9	21	(2%)
10	10	(1%)
11	72	(8%)
12	45	(5%)
13	37	(4%)
14	38	(4%)
15	30	(4%)
16	57	(7%)
17	36	(4%)

Por fim executei o *DBSCAN* com os resultados apresentados abaixo. Verifica-se que cria, desta vez, dois clusters.

```
=== Model and evaluation on training set ===
```

```
Clustered Instances
```

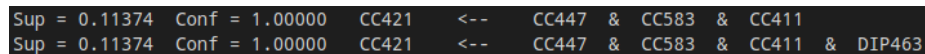
0	233	(28%)
1	612	(72%)

2 Considere o dataset de disciplinas (student_courses.bas, descarregar do blackboard). Apresente para este dataset exemplos de regras redundantes, produtivas e não produtivas e regras significativas. Comente os exemplos obtidos/usados

Corri o seguinte comando no terminal de modo a obter todas as regras para o modelo.

```
$ java caren ../student_courses.bas 0.1 0.5 -s, -d
```

Consegue-se verificar **regras redundantes**, isto é, existem regras que são anteriormente referidas por outras regras. Como exemplo demonstrado na seguinte figura.



```
Sup = 0.11374  Conf = 1.00000  CC421  <--  CC447  &  CC583  &  CC411
Sup = 0.11374  Conf = 1.00000  CC421  <--  CC447  &  CC583  &  CC411  &  DIP463
```

Figura 1: Output

De seguida, adicionei o campo *-imp0.01* para obter as regras maiores ou iguais a 0.

```
$ java caren ../student_courses.bas 0.1 0.5 -s, -d -imp0.01
```

Obtêm-se assim as **regras produtivas**, isto é, as regras com o valor de *improvement* maior que 0, uma mais valia ao nível da medida de interesse. verifica-se, agora, que as linhas output que eram 36000 agora são só 3000.

Por fim usei a opção *-fisher*. Esta opção rejeita hipóteses nulas.

```
$ java ../caren student_courses.bas 0.1 0.5 -s, -d -fisher
```

desta vez obteve-se apenas 700 linhas de output. Ou seja as **regras significativas**, aquelas que apresentam mais confiança em todas as generalizações.

- 3 Considere o dataset adult (descarregar do blackboard). Usando o sistema CAREN e os seus vários métodos para Subgroup Mining apresente e comenta regras que denunciam discriminação de género e idade em termos de rendimentos anuais. Notar: o atributo classe deste dataset Adult caracteriza os rendimentos anuais dos indivíduos registados i.e. $\leq 50K$ indica rendimento anual menor que 50 000 USD (e o seu oposto).

Para responder a esta pergunta executei o seguinte comando que cria um ficheiro *t.txt* com as regras para posterior análise.

```
java -cp . caren ../adult.wd-fi.csv.test 0.01 0.5 -s, -Att  
-h"class=<=50K", "class=>50K" -CS -ovrt -null"?"
```

Usando a funcionalidade do *vscode* para procurar usando uma expressão regular $((class \Rightarrow 50K \gg class \leq 50K)114 \backslash n. * (sex|age). * (sex|age).*)$ obtemos 10 match, que que denunciam discriminação de género e idade em termos de rendimentos anuais.