Universidade do Minho
Escola de Engenharia

Mineração de Dados

Exercício Prático nº 3 - Individual
MiEI - 4º Ano - 1º Semestre

A84656 Hugo Manuel Cunha

Braga,
6 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.

Começando por selecionar a classe *Outcome* para avaliação os algoritmos vão ignorar este atributo para o processo de classificação. Correndo o algoritmo **EM** no dataset "Unbalanced"obtemos 18 partições. Destes 18 apenas o Cluster 2 foi considerado como representativo da classe *Inactive* e o Cluster 13 da classe *Active* o que leva a uma avaliação de Incorrectly Classified Instances de 84%. Este algoritmo compara a probabilidade de um elemento pertencer a um cluster para o selecionar.
É de referir que este algoritmo não considera outliers e agrupa em clusters todos os dados do dataset.

```
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 %
```

(a) Resultado do algoritmo EM

Usando o algoritmo K-means com 18 clusters obtemos resultados semelhantes com dois deles a serem representativos de cada classe avaliando com 85% de ICI. Este algoritmo também nos diz que a soma erro quadrado dentro dos clusters é de 321. Este algoritmo foi muito mais rápido na sua execução (0.1 segundos) em comparação com o algoritmo **EM** (70 segundos) Sobre o algoritmo podemos prever que devido á restrição de usar uma medida de distancia vamos obter clusters esféricos em todas as dimensões podendo ter um erro superior a outros algoritmos mais complexos.

```
Classes to Clusters:

   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  <-- assigned to cluster
   0   0   0   0   0   0   0   2   1   0   0   2   3   1   2   1   0   0 | Active
  22  19  75  27  52 122  39 100  51  21  10  70  42  36  36  29  57  36 | Inactive

Cluster  0 <-- No class
Cluster  1 <-- No class
Cluster  2 <-- No class
Cluster  3 <-- No class
Cluster  4 <-- No class
Cluster  5 <-- Inactive
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 <-- Active
Cluster 13 <-- No class
Cluster 14 <-- No class
Cluster 15 <-- No class
Cluster 16 <-- No class
Cluster 17 <-- No class

Incorrectly clustered instances :       731.0    85.3972 %
```

(a) Resultado do algoritmo Simple K-means

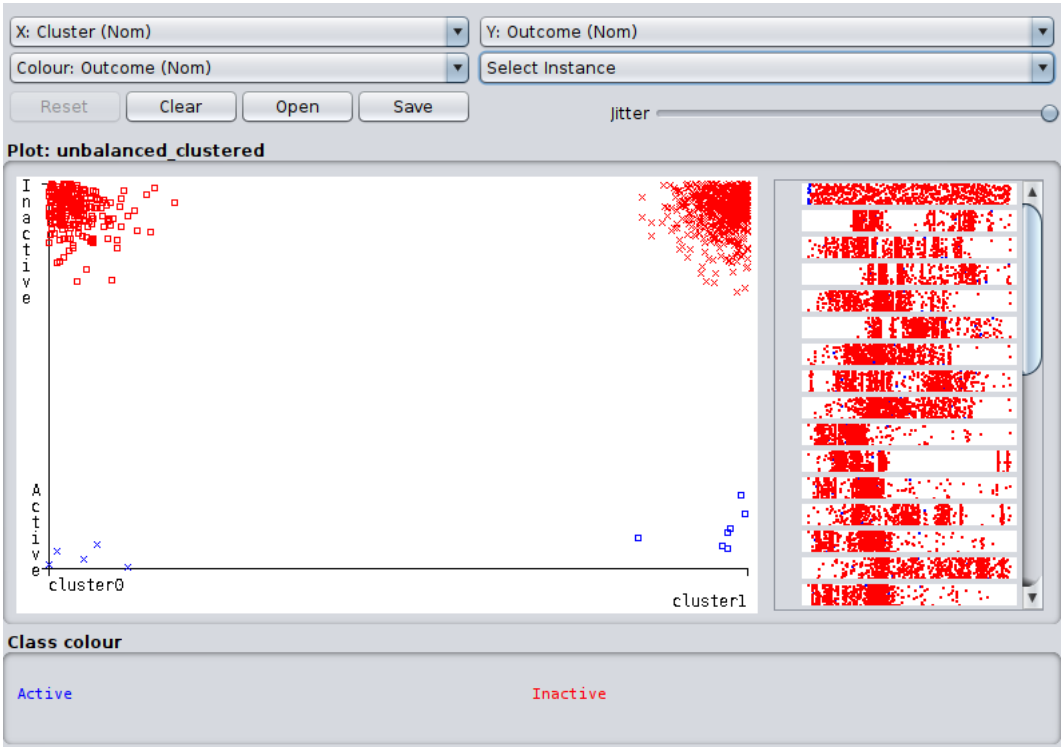
O algoritmo **DBSCAN** corre em apenas 0.34 segundos no entanto apenas produz 2 clusters. Associando cada um a uma classe obtemos um ICI de 27.45%. Este algoritmo detetou 11 instancias de Outliers que depois de analizar os testes de cada instância e comparar com o ficheiro de dados posso confirmar que todos são da classe **Inactive** ou seja, nenhum dos escassos elementos da classe **Active** foi descartado, facto importante a verificar quando tratamos de um dataset tão desbalanceado.

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

Clustered Instances

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

Unclustered instances : 11

Class attribute: Outcome
Classes to Clusters:

   0   1  <-- assigned to cluster
   5   7 | Active
 228 605 | Inactive

Cluster 0 <-- Active
Cluster 1 <-- Inactive

Incorrectly clustered instances :       235.0    27.4533 %
```

(a) Resultado do algoritmo DBSCAN

Podemos confirmar que também neste dataset o ICI não é uma boa medida pois no gráfico de classe vs cluster vê-se que não não há uma atribuição evidente de um cluster a uma classe.

Este algoritmo agrupa os dados consoante a densidade dos elementos ao longo do dominio dos atributos pelo que deriva a quantidade de clusters necessários.



(a) Gráfico dos clusters DBSCAN

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

Ao correr o comando "java caren ../student_courses.bas 0.1 0.5 -s, -d ¿ ../allRules.txt"obtemos todas as regras derivadas do dataset de disciplinas e guarda num ficheiro para comparar mais tarde.
Esta lista de regras contém **Regras Redundantes** como:

```
51  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411
52  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  DIP463
53  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  CC422
54  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  CC420
55  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  GEST400
56  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  DIP461
57  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  CC422  &  DIP463
58  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  CC420  &  DIP463
59  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  CC420  &  CC422
60  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  GEST400  &  DIP463
61  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  GEST400  &  CC422
62  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  GEST400  &  CC420
63  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  DIP461  &  DIP463
64  Sup = 0.11374  Conf = 1.00000   CC421   <--   CC447  &  CC583  &  CC411  &  DIP461  &  CC422
```

(a) Regras redundantes

Uma **regra redundante** é uma regra que tem generalizações com o mesmo suporte, assim se cortarmos as mais específicas podemos diminuir a árvore aumentando a velocidade de processamento de casos de teste. Neste caso, as regras assinaladas são redundantes pois acrescentam complexidade á regra de cima (linha 51) sem alterar o suporte.

Regras produtivas são regras que aumentam uma medida em relação ás suas mais genéricas, no *CAREN* o improvement é calculado pela confiança. Usei o comando "java caren ../student_courses.bas 0.1 0.5 -s, -d -imp0.01 ¿ ../improvRules.txt"para obter todas as regras filtradas para apresentar só as regras um improvement de confiança superior a 0.01 .

```
27  Sup = 0.11374  Conf = 1.00000  CC421  <--  CC447 & CC583 & CC411
28  Sup = 0.10900  Conf = 1.00000  CC421  <--  CC447 & CC583 & CC413
29  Sup = 0.10900  Conf = 1.00000  CC421  <--  CC447 & CC583 & CC412
```

(a) Regra produtiva

Aqui podemos ver que as regras redundantes apresentadas anteriormente não são produtivas pois têm a mesma confiança que a regra mais genérica que é uma **regra produtiva**.

As **regras significativa** são regras cujo improvement é estatisticamente significante, isto é, em vez de definir um improvement mínimo usamos um teste estatístico. Podemos usar o teste de fisher no **CAREN** para filtrar as regras não significantes.

```
24  Sup = 0.12796  Conf = 1.00000  CC421  <--  CC447 & CC583 & DIP463
25  Sup = 0.12796  Conf = 1.00000  CC421  <--  CC447 & CC583 & CC420
26  Sup = 0.17062  Conf = 0.97297  CC421  <--  CC583 & CC422
27  Sup = 0.12796  Conf = 0.96429  CC421  <--  CC447 & CC583
28  Sup = 0.17536  Conf = 0.94872  CC421  <--  CC410 & CC422
```

(a) Regra significante

Aqui podemos ver que a regra produtiva observada anteriormente não foi considerada significativa embora regras muito parecidas sejam. Esta foi substituída pela sua mais genérica. (linha 27)

Neste exercico aplicamos duas formas de pruning diferentes que reduziram o rule set inicial de 36600 para 2400 com pruning de **regras não produtivas** passando para 6% e para 670 usando pruning de **regras não significativas** passando para apenas 2%.

Isto diminui a quantidade de regras de um sistema de previsão o que permite não só gastar menos armazenamento como também menos recursos para processar testes sobre o sistema.

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. <=50K indica rendimento anual menor que 50 000 USD(e o seu oposto).

Para fazer *Subgroup Mining* com **CAREN** usei o comando "caren ../adult.wdfi.csv.test 0.01 0.5 -s, -Att -hclass=<=50K,class=>50K -CS -ovrt -null?"que gerou um ficheiro t.txt dos quais se pode obter os resultados.

Para obter a lista de regras que mencionam a idade e o sexo utilizei a funcionalidade de procurar texto com um regex do *VSCode* com o regex :

```
^.*\n.*( age|sex).*( age|sex).*$
```

Copiei para um ficheiro de texto as 7 regras encontradas e apaguei as 4 regras mais específicas deixando apenas 3:

```
Obs =  000360 | 000485  Gsup  = 0.09360 | 0.03900  p = 4.2307999064E-036  phi =  0.10455   class=>50K >> class=<=50K
Sup(CS) =  0.05190                                                              <---     age=(54.5-61.5]  &  sex=Male
Obs =  000668 | 001109  Gsup  = 0.17369 | 0.08918  p = 7.9856054182E-045  phi =  0.11511   class=>50K >> class=<=50K
Sup(CS) =  0.10915                                                              <---     age=(35.5-41.5]  &  sex=Male
Obs =  001320 | 001528  Gsup  = 0.34321 | 0.12288  p = 7.1944316838E-194  phi =  0.24635   class=>50K >> class=<=50K
Sup(CS) =  0.17493                                                              <---     age=(41.5-54.5]  &  sex=Male
```

(a) Discriminação de género e idade

Aqui podemos ver que individuos do sexo masculino com idade entre 35.5 e 61.5 anos têm uma probabilidade de fazer parte da classe salarial $>= 50K\$$ por ano muito superior ao contrário.