**WYŻSZA SZKOŁA INFORMATYKI**

**I ZARZĄDZANIA**

Z SIEDZIBĄ W RZESZOWIE

***Algorytm generowania reguł LEM2***

Prowadzący: dr inż. Mariusz Wrzesień Wykonawca: Oleh Korsunskyi w58957

Rzeszów 2019

# Opis problemu

Celem laboratorium jest utworzenie zestawu reguł decyzyjnych, dla danych umiejscowionych w tablicy decyzji, przy pomocy algorytmu generowania reguł LEM2.

# Badane zbiory danych

Dane, jakie należy przetworzyć, dotyczą owoców. Są one przedstawione za pomocą tablicy decyzji, która składa się z sześciu kolumn i dwudziestu jednego wierszu. Poszczególne wiersze reprezentują cechowy opis przypadków(obiektów). Kolumna pierwsza przedstawia atrybut nominalny o nazwie „color”. Zawiera cztery wartości atrybutów: green, red, maroon, yellow. Kolumna druga przestawia atrybut porządkowy „size”. Posiada trzy wartości, medium, small, big. Trzecia kolumna przedstawia atrybut binarny „shape” i dysponuje dwoma wartościami, round i elongated. Kolumna numer 4 o atrybucie binarnym ,,taste” posiada także dwie wartości sour oraz sweet. Przedostatnia kolumna przedstawia atrybut liczbowy „weight” wraz z pięcioma wartościami, 0.1000, 0.2000, 0.3000, 0.4000, 0.5000. Ostatnia kolumna jest kolumną decyzji „fruit” . Składa się z sześciu klas: Lemon, Apple, Grapefruit, Grape, Banana, Cherry.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| LP | color | size | shape | taste | weight | fruit |
| 1 | yellow | big | round | sour | 0.6000 | Grapefruit |
| 2 | green | medium | round | sweet | 0.5000 | Apple |
| 3 | green | medium | round | sour | 0.1000 | Lemon |
| 4 | red | medium | round | sweet | 0.3000 | Apple |
| 5 | green | medium | round | sour | 0.1000 | Lemon |
| 6 | red | small | round | sweet | 0.1000 | Cherry |
| 7 | green | medium | round | sour | 0.3000 | Apple |
| 8 | red | medium | round | sweet | 0.4000 | Apple |
| 9 | green | big | round | sour | 0.5000 | Grapefruit |
| 10 | green | small | round | sweet | 0.1000 | Grape |
| 11 | red | medium | round | sour | 0.4000 | Apple |
| 12 | maroon | small | round | sweet | 0.2000 | Grape |
| 13 | green | small | round | sour | 0.1000 | Grape |
| 14 | green | medium | elongated | sweet | 0.2000 | Banana |
| 15 | yellow | medium | elongated | sweet | 0.3000 | Banana |
| 16 | yellow | big | elongated | sweet | 0.3000 | Banana |
| 17 | green | big | round | sweet | 0.4000 | Apple |
| 18 | green | medium | round | sour | 0.1000 | Lemon |
| 19 | yellow | medium | round | sour | 0.2000 | Lemon |
| 20 | yellow | small | round | sour | 0.1000 | Lemon |
| 21 | red | big | round | sweet | 0.5000 | Apple |

# Obliczenia

Działanie algorytmu LEM 2 przedstawię na przykładzie regulowania reguł dla jednej klasy. Poniższy opis przedstawia sposób generowania reguł dla klasy Grape.

Pierwszym krokiem jest wyznaczenie zbioru B, który zawiera wszystkie przypadki klasy Grape:

***B= {10, 12, 13}***

Następnie zostaje wyznaczony zbiór T(G) wszystkich możliwych par (atrybut,wartość) spośród przypadków ze zbioru B:

***T(G) = {(color, green), (color, maroon), (size, small), (shape, round), (taste, sweet), (taste, sour), (weight, 0.1), (weight, 0.2)}***

Wstępnie założono, że zbiór ***B = G = {10, 12, 13} oraz n = Ø.***

Dalej wypisuje wszystkie możliwe przypadki z tablicy decyzji, które należą do danej kombinacji (atrybut, wartość). Przypadki dotyczące klasy Grape zostały zapisane w pierwszych nawiasach, a przypadki z innych klas w drugich nawiasach.

|  |
| --- |
| (color, green) = {10, 13}, {2, 3, 5, 7, 9, 14, 17, 18} |
| (color, maroon) = {12} |
| (size, small) = {10, 12, 13}, {6, 20} |
| (shape, round) = {10, 12, 13}, {1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 17, 18, 19, 20, 21} |
| (taste, sweet) = {10, 12}, {2, 4, 6, 8, 14, 15, 16, 17, 21} |
| (taste, sour) = {13}, {1, 3, 5, 7, 9, 11, 18, 19, 20} |
| (weight, 0.1) = {10, 13}, {3, 5, 6, 18, 20} |
| (weight, 0.2) = {12}, {14, 19} |

Najważniejszym parametrem wyboru par (atrybut wartość) jest część wspólna ze zbiorem G, im większa tym lepsza. Dodatkowym parametrem jest moc, w tym przypadku im mniejsza tym lepiej.

Najwięcej przypadków z klasy Grape zawierają kombinacje (size, small) oraz (shape, round) (po trzy przypadki). Algorytm w pierwszej kolejności wybiera zestawienie (size, small), ponieważ pokrywa wszystkie przypadki klasy Grape i ma większą moc. Warunek pokrywa także dwa elementy z innych klas obiekty {6, 20}, dlatego w kolejnym kroku należy dobrać dodatkowy warunek, który wyeliminuje przypadki, które są nie z tej klasy.

Poszukując kolejnej kombinacji, nie uwzględniamy juz w zbiorze T(G) warunku (size, small) i dobieramy kolejną parę, która ma najwięcej przypadków klasy Grape i największą moc. Jest to para (shape, round). Dodanie jej do warunku nie utworzy reguły, ponieważ nadal są pokryte, obiekty innych klas.

Następnie dodajemy zgodnie z algorytmem jeszcze jedną parę (weight, 0.1), której nie

również nie wystarcza, żeby stworzyć regułę. Dodajemy jeszcze jedną parę (color, green) i teraz możemy stworzyć regułę.

(T = {(color, green), (shape, round), (weight, 0.1000), (size, small)}, więc [T] = {10, 13}) oraz [T] ⊂ B można utworzyć pierwszą, następującą regułę:

***JEZELI color = green ORAZ size = small ORAZ shape = round ORAZ weight = 0.1 TO fruit JEST Grape***

G = B - [T] = {12}

Klasa Grape składa się z trzech przypadków, natomiast powyższa reguła wyznaczyła tylko dwa z nich. Dlatego następnym krokiem jest wyliczenie reguły pokrywającej pozostały przypadek, postępując w ten sam sposób jak wyżej.

Na nowo zostają wyszukane w zbiorze T(G) kombinacje z największą ilością obiektów należących do zbioru G= {12}:

|  |
| --- |
|  |
| (color, maroon) = {12} |
| (size, small) = {12}, {6, 20, 10, 13} |
| (shape, round) = {12}, {1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 17, 18, 19, 20, 21, 10, 13} |
| (taste, sweet) = {12}, {2, 4, 6, 8, 14, 15, 16, 17, 21, 10} |
| (weight, 0.2) = {12}, {14, 19} |
|  |

Zgodnie z algorytmem wybieramy parę (color, maroon), i tego atrybutu wystarcza, że pokryć pozostały przypadek

W wyniku połączenia warunków otrzymano:

T= {12}

Ponieważ spełniony jest warunek [T] ⊂ B można utworzyć regułę 2:

***JEZELI color = maroon TO fruit JEST Grape***

G = B - [T] = {10, 12, 13} - {10, 12, 13} = Ø

G = Ø zatem wszystkie przypadki z kategorii Grape zostały pokryte przez reguły. Algorytm kończy działanie dla tej klasy.

# Wyniki

W wyniku opisanego powyżej algorytmu powstało dwanaście reguł pokrywające wszystkie przypadki:

|  |  |
| --- | --- |
| REGULA 1 | JEZELI size = big ORAZ taste = sour TO fruit JEST Grapefruit |

Przypadki: {1, 9}

|  |  |
| --- | --- |
| REGULA 2 | JEZELI size = medium ORAZ shape = round ORAZ taste = sweet TO fruit JEST Apple |

Przypadki: {2, 4, 8}

|  |  |
| --- | --- |
| REGULA 3 | JEZELI shape=round ORAZ weight = 0.4 TO fruit JEST Apple |

Przypadki: {11, 17}

|  |  |
| --- | --- |
| REGULA 4 | JEZELI shape = round ORAZ color = red ORAZ weight = 0.5 TO fruit JEST Apple |

Przypadki: {21}

|  |  |
| --- | --- |
| REGULA 5 | JEZELI taste = sour ORAZ weight = 0.3 TO fruit JEST Apple |

Przypadki: {7}

REGULA 6 JEZELI size = medium ORAZ shape = round ORAZ taste = sour ORAZ weight = 0.1 TO fruit JEST Lemon

Prypadki: {3, 5, 18}

REGULA 7 JEZELI color = yellow ORAZ shape = round ORAZ taste = sour ORAZ weight = 0.2 TO fruit JEST Lemon

Przypadki: {19}

|  |  |
| --- | --- |
| REGULA 8 | JEZELI color = yellow ORAZ size = small TO fruit jest Lemon |

Przypadki: {20}

|  |  |
| --- | --- |
| REGULA 9 | JEZELI color = red ORAZ size = small TO fruit JEST Cherry |

Przypadki: {6}

REGULA 10 JEZELI color = green ORAZ size = small ORAZ shape = round ORAZ weight = 0.1 TO fruit JEST Grape

Przypadki: {10, 13}

REGULA 11 JEZELI color = maroon TO fruit JEST Grape

Przypadki: {12}

REGULA 12 JEZELI shape = elongated To fruit JEST Banana

Przypadki: {14, 15, 16}

# Wnioski

Algorytm LEM2 jest wykorzystywany dla generowania reguł decyzji na podstawie bazy informacyjnej. Do tego celu wykorzystuje teorię zbiorów przybliżonych. Reguły wyszukiwane są w sposób iteracyjny, a wynik to zbiór reguł, które pokrywają wszystkie przypadki ze zbioru uczącego.

Z powyższej bazy wiedzy, przy pomocy algorytmu LEM2, powstało dwanaście reguł pokrywających wszystkie przypadki. Reguła dwunasta, dotycząca klasy Banana pokrywa wszystkie przypadki swojej kategorii, natomiast, na przykład, aby pokryć klasę Apple algorytm musiał stworzyć aż cztery reguły. Tak się dzieje, ponieważ dla tego algorytmu najważniejsza jest wspólna cześć pary ze zbiorem G, a moc jest tylko parametrem dodatkowym. W przykładzie jest para (shape, round], która zawiera aż 85% przypadków. W trakcie tworzenia reguł algorytm bierze tę parę pod uwagę, choć tak naprawdę nic ona nie wnosi do reguły. Dodawanie w trakcie tworzenia reguł zbędnych jest największą wadą LEM2.