Integrating Classification and Association Rule Mining
— the Secret Behind CBA

Written by Bing Liu, etc.

CBA Advantages

- One algorithm performs 3 tasks
- It can find some valuable rules that existing classification systems cannot.
- It can handle both table form data and transaction form data
- It doesn’t require the whole database to be fetched into the main memory.
Problem Statement

Classification (predetermined target) +

Association (no fix targets) →

CBA (Classification Based on Associations)

Input and Output

- **Input**
  - Table form dataset (transformed needed) or transaction form dataset.

- **Output**
  - A complete set of CARs, (class association rule) – done by CBA-RG (rule generator)
  - A classifier. – done by CBA-CB (classifier builder)
CBA-RG: Basic concepts (1)

- The key operation of CBA-RG is to find all ruleitems that have support above minsup.
- **ruleitem**: \(<\text{condset}, y>\), representing the rule: condset \(\rightarrow y\)
- **condsupCount**: # of cases in D that contain the condset.
- **rulesupCount**: # of cases in D that contain the condset and are labeled with class y.

CBA-RG: Basic concepts (2)

- **support**: \((\text{rulesupCount} / |D|) * 100\%\).
- **confidence**: \((\text{rulesupCount} / \text{condsupCount}) * 100\%\)

Example:
- Ruleitem: \(<\{(A, e), (B, p)\}, (C, y)\>
- condsupCount: 3
- rulesupCount: 2
- support: \((2 / 10) * 100\% = 20\%\)
- confidence: \((2 / 3) * 100\% = 66.7\%\)
CBA-RG: Basic concepts (3)

- **k-ruleitem**: A ruleitem whose condset has k items.
- **frequent ruleitems**: Ruleitems that satisfy minsup. Denoted as $F_k$ in the algorithm.
- **candidate ruleitems**:
  - Possibly frequent ruleitems generated somehow from the frequent ruleitems found in the last pass. Denoted as $C_k$.
- A ruleitem is represented in the algorithm in the form:
  - $<(\text{condset}, \text{condsupCount}), (y, \text{rulesupCount})>$

The CBA-RG algorithm

```java
1 $F_1 = \{ \text{large 1-ruleitems} \}$;
2 $CAR_1 = \text{genRules}(F_1)$;
3 $prCAR_1 = \text{pruneRules}(CAR_1)$;
4 for ($k = 2; F_i \neq \emptyset; k++$) do
5   $C_k = \text{candidateGen}(F_{i-1})$;
6   for each data case $d \in D$ do
7     $C_d = \text{ruleSubset}(C_k, d)$;
8     for each candidate $c \in C_d$ do
9       $c.\text{condsupCount}++$;
10      if $d.\text{class} = c.\text{class}$ then $c.\text{rulesupCount}++$
11    end
12 end
13 $F_i = \{ c \in C_i | c.\text{rulesupCount} \geq \text{minsup} \}$;
14 $CAR_i = \text{genRules}(F_i)$;
15 $prCAR_i = \text{pruneRules}(CAR_i)$;
16 end
17 $CAR_S = \bigcup_i CAR_i$;
18 $prCAR_S = \bigcup_i prCAR_i$;
```
A case study

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>p</td>
<td>y</td>
</tr>
<tr>
<td>e</td>
<td>p</td>
<td>y</td>
</tr>
<tr>
<td>e</td>
<td>q</td>
<td>y</td>
</tr>
<tr>
<td>g</td>
<td>q</td>
<td>y</td>
</tr>
<tr>
<td>g</td>
<td>q</td>
<td>n</td>
</tr>
<tr>
<td>g</td>
<td>w</td>
<td>n</td>
</tr>
<tr>
<td>g</td>
<td>w</td>
<td>n</td>
</tr>
<tr>
<td>e</td>
<td>p</td>
<td>n</td>
</tr>
<tr>
<td>f</td>
<td>q</td>
<td>n</td>
</tr>
</tbody>
</table>

**Attributes:** A, B

**Class:** C

**minsup:** 15%

**minconf:** 60%

---

1st pass

**F1**

\(<((A, e), 4), ((C, y), 3)>, <((A, g), 5), ((C, y), 2)>, <((A, g), 5), ((C, n), 3)>, <((B, p), 3), ((C, y), 2)>, <((B, q), 5), ((C, y), 3)>, <((B, q), 5), ((C, n), 2)>, <((B, w), 2), ((C, n), 2)>

2nd pass

**C2**

\(<((A, e), (B, p)), (C, y)>, <((A, e), (B, q)), (C, y)>, <((A, g), (B, p)), (C, y)>, <((A, g), (B, q)), (C, y)>, <((A, g), (B, q)), (C, n)>, <((A, g), (B, w)), (C, n)>

**F2**

\(<((A, e), (B, p)), 3), ((C, y), 2)>, <((A, g), (B, q)), 3), ((C, y), 2)>, <((A, g), (B, q)), 3), ((C, n), 1)>, <((A, g), (B, w)), 2), ((C, n), 2)>

**CAR1**

\((A, e) \rightarrow (C, y), (A, g) \rightarrow (C, n), (B, p) \rightarrow (C, y), (B, q) \rightarrow (C, y), (B, w) \rightarrow (C, n)\)

**CAR2**

\(((A, e), (B, p)) \rightarrow (C, y), ((A, g), (B, q)) \rightarrow (C, y), ((A, g), (B, w)) \rightarrow (C, n)\)

**CARs**

CAR1 ∪ CAR2
genRules(Fk):
• **possible rule** (PR): For all the ruleitem that have the same condset, the ruleitem with the highest confidence is chosen as a PR.
• If there are more than one ruleitem with the same highest confidence, we randomly pick one.
• **accurate rule**: confidence >= minconf

pruneRules(CARk):

<table>
<thead>
<tr>
<th>prCAR1</th>
<th>(A, e)→(C, y), (A, g)→(C, n), (B, p)→(C, y), (B, q)→(C, y), (B, w)→(C, n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>prCAR2</td>
<td>{(A, g), (B, q)} → (C, y)</td>
</tr>
<tr>
<td>prCARs</td>
<td>prCAR1 ∪ prCAR2</td>
</tr>
</tbody>
</table>

Classifier Builder

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>p</td>
<td>y</td>
</tr>
<tr>
<td>e</td>
<td>p</td>
<td>y</td>
</tr>
<tr>
<td>e</td>
<td>q</td>
<td>y</td>
</tr>
<tr>
<td>g</td>
<td>q</td>
<td>y</td>
</tr>
<tr>
<td>g</td>
<td>q</td>
<td>n</td>
</tr>
<tr>
<td>g</td>
<td>w</td>
<td>n</td>
</tr>
<tr>
<td>g</td>
<td>w</td>
<td>n</td>
</tr>
<tr>
<td>c</td>
<td>p</td>
<td>n</td>
</tr>
<tr>
<td>f</td>
<td>q</td>
<td>n</td>
</tr>
</tbody>
</table>

CARs after pruning:
(1) A = e → y  sup=3/10  conf=3/4
(2) A = g → n  sup=3/10  conf=3/5
(3) B = p → y  sup=2/10  conf=2/3
(4) B = q → y  sup=3/10  conf=3/5
(5) B = w → n  sup=2/10  conf=2/2
(6) A = g, B = q → y  sup=2/10  conf=2/3
CBA-classifier builder

- **Goal**: select a small set of rules from the complete CARs as the classifier

\[<r_1, r_2, \ldots, r_n, \text{default\_class}>\]

where \( r_i \in R \), \( r_a \succ r_b \) if \( b \succ a \). default\_class is the default class.

CBA-CB specification

- \( \succ \) (Precedence) definition
  Given two rules, \( r_i \) and \( r_j \), \( r_i \succ r_j \) (also called \( r_i \) precedes \( r_j \) or \( r_i \) has a higher precedence than \( r_j \)) if
  1. the confidence of \( r_i \) is greater than that of \( r_j \), or
  2. their confidences are the same, but the support of \( r_i \) is greater than that of \( r_j \) or
  3. both the confidences and supports of \( r_i \) and \( r_j \) are the same, but \( r_i \) is generated earlier than \( r_j \).
CBA-CB two algorithms

- Two algorithms
  - **M1** (the database can be fetched into and processed in main memory). Suitable for **small datasets**
  - **M2** (the database can be resident in hard disk.) suitable for **huge datasets**

CBA-CB satisfaction conditions

- **Two conditions**
  - **Condition 1.** Each training case is covered by the rule with the highest precedence among the rules that can cover the case.
  - **Condition 2.** Every rule in C correctly classifies at least one remaining training case when it is chosen.
CARs after pruning:

(1) A = e → y  sup=3/10  conf=3/4
(2) A = g → n  sup=3/10  conf=3/5
(3) B = p → y  sup=2/10  conf=2/3
(4) B = q → y  sup=3/10  conf=3/5
(5) B = w → n  sup=2/10  conf=2/2
(6) A = g, B = q → y  sup=2/10  conf=2/3

<table>
<thead>
<tr>
<th>rule</th>
<th>#covCases</th>
<th>#eCovered</th>
<th>#wCovered</th>
<th>defClass</th>
<th>#errors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1  \( R = \text{sort}(R); \)
2  for each rule \( r \in R \) in sequence do
3     \( \text{temp} = \emptyset; \)
4     for each case \( d \in D \) do
5         if \( d \) satisfies the conditions of \( r \) then
6             store \( d \) id in \( \text{temp} \) and mark \( r \) if it correctly classifies \( d \);
7         if \( r \) is marked then
8             insert \( r \) at the end of \( C \);
9             delete all the cases with the ids in \( \text{temp} \) from \( D \);
10        selecting a default class for the current \( C \);
11        compute the total number of errors of \( C \);
12     end
13 end
14 Find the first rule \( p \) in \( C \) with the lowest total number of errors and drop all the rules after \( p \) in \( C \);
15 Add the default class associated with \( p \) to end of \( C \), and return \( C \) (our classifier).
CBA-CB M2

- M2 (more efficient algorithm for large datasets)

**Key point:** instead of making one pass over the remaining data for each rule (in M1), we find the best rule in R to cover each case.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>covRules</th>
<th>cRule</th>
<th>wRule</th>
<th>U</th>
<th>Q</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>p</td>
<td>y</td>
<td>1,3</td>
<td>null</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>p</td>
<td>y</td>
<td>1,3</td>
<td>null</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>q</td>
<td>y</td>
<td>1,3</td>
<td>null</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>g</td>
<td>q</td>
<td>y</td>
<td>2,4,6</td>
<td>6</td>
<td>2</td>
<td>1.6</td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>g</td>
<td>q</td>
<td>n</td>
<td>2,4,6</td>
<td>6</td>
<td>2</td>
<td>1.6</td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>g</td>
<td>w</td>
<td>n</td>
<td>2,4,6</td>
<td>6</td>
<td>2</td>
<td>1.6</td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>p</td>
<td>n</td>
<td>1,3</td>
<td>null</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>q</td>
<td>n</td>
<td>4</td>
<td>null</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

CARs after pruning:

1. A = e → y sup=3/10 conf=3/4
2. A = g → y sup=2/10 conf=2/3
3. B = p → n sup=3/10 conf=3/5
4. B = q → y sup=3/10 conf=3/5
5. B = w → n sup=2/10 conf=2/2
6. A = g, B = q → y sup=2/10 conf=2/3
Empirical Evaluation

- 26 datasets from UIC ML Repository
- The results show that CBA produces more accurate classifiers.
- On average, the error rate decreases from 16.7% for C4.5rules to 15.6%-15.8% for CBA.
- Without or with rule pruning the accuracy of the resultant classifier is almost the same. So, the prCARs are sufficient for building accurate classifiers.
- Experiments show that both CBA-RG and CBA-CB(M2) have linear scaleup.

Conclusion

- Proposing a framework to integrate classification and association rule mining.
- An algorithm that generate all class association rules (CARs) and to build an accurate classifier.
- Contributions:
  - A new way to construct accurate classifiers;
  - It makes association rule mining techniques applicable to classification tasks;
  - It helps to solve a number of questions existing in current classification systems.