Association Rule Mining: MiniSat and Apriori

Laura Iacovissi

University of Padova - Dipartimento di Matematica

Introduction

The goal of this study is to implement a propositional satisfiability based algorithm for Association Rule Mining (ARM), comparing its results with the ones obtained using Apriori.

The structure of the logical model is inspired by the article *"A SAT-Based Approach for Mining Association Rules"* (Abdelhamid Boudane, Said Jabbour, Lakhdar Sais, Yakoub Salhi, 2016).

The dataset used to test the algorithm is the *"1984 United States Congressional Voting Records Data Set"*, available on the UCI Machine Learning Repository platform.

The SAT-based algorithm is implemented using the `PySAT` package, the Apriori-based one using `efficient_apriori`.

Data Preparation

The original dataset includes votes for each of the U.S. House of Representatives Congressmen on 16 key topics (yes, no, unknown). For each Congressmen is specified the political trend, republican or democratic, in the Class Name attribute.

The dataset has been modified adding the Republican and the Democrat attributes.

After that, two kind of paths has been followed to deal with missing values:

- rows with unknown values have been dropped and only positive votes have been considered as items (Y);
- all the data have been kept but the set of items contains both positive and negative votes (Y-N).

SAT encoding for ARM - Definitions

- Ω : finite set of items
  $D = (i, I_i)_{i=1}^{n}$: transaction database, $i \in \mathbb{N}, I_i \subset \Omega$
  α : minimum support threshold
  β : minimum confidence threshold

- An association rule has the form $X \to Y$, with $X, Y \subset \Omega$ disjoint itemsets

- $S(A, D) = |\{i \in \mathbb{N} s.t.\ A \subset I_i, (i, I_i) \in D\}|\ ,\ A \subset \Omega$

  $S(X \to Y, D) = \frac{S(X \cup Y, D)}{|D|}$

  $Conf(X \to Y, D) = \frac{S(X \cup Y, D)}{S(X, D)}$

- x_a : propositional variable, true if the item $a \in \Omega$ is in X
  y_a : propositional variable, true if the item $a \in \Omega$ is in Y
  p_i : propositional variable, true if $X \subset I_i$
  q_i : propositional variable, true if $X \cup Y \subset I_i$

SAT encoding for ARM - Rules

Rule 1 $\bigwedge_{a\in\Omega}(\neg x_a \vee \neg y_a)$: X and Y disjoint

Rule 2 $\bigwedge_{i=1}^{|D|}(\neg p_i \leftrightarrow \bigvee_{a\in\Omega\setminus I_i} x_a)$: characterization of p_i

Rule 3 $\bigwedge_{i=1}^{|D|}(\neg q_i \leftrightarrow \neg p_i \vee (\bigvee_{a\in\Omega\setminus I_i} y_a))$: characterization of q_i

Rule 4 $\sum_{i=1}^{|D|} q_i \geq |D| \times \alpha$

Rule 5 $\frac{\sum_{i=1}^{|D|} q_i}{\sum_{i=1}^{|D|} p_i} \geq \beta$

Plus two additional rules, not in the paper:

Rule 6 $\bigvee_{a\in\Omega} x_a \wedge \bigvee_{a\in\Omega} y_a$: at least one item in each set of the rule

Rule 7 $\bigwedge_{a,b\in\Omega,a\neq b}(\neg y_a \vee \neg y_b)$: minimality condition for Y, at most one element

Pseudo-Boolean Rules

Rule 4 and 5 are Pseudo-Boolean conditions, i.e. a condition of the form

$$\sum_{i=1}^{n} w_i x_i * k$$

where w_i are integer weights, x_i literals, k an integer bound and $* \in \{=, \leq, \geq\}$. They express the frequency (support) and confidence constraints.

These two rules are managed through the `PBEnc` class. In particular, rule 5 can be rewritten defining

$$w_i = 100\ \mathbf{1}_{\{i \leq |D|\}} - 100\beta\ \mathbf{1}_{\{i > |D|\}}$$

$$x_i = q_i\ \mathbf{1}_{\{i \leq |D|\}} + p_{i-|D|}\ \mathbf{1}_{\{i > |D|\}}$$

for $i = 1, \ldots, 2|D|$.

Rule extraction with MiniSat

Due to the huge number of hidden variables generated from the Pseudo-Boolean constraints CNF encoding, the full set of extracted models is large and contains repetitions.

Trying to avoid this problem, the algorithm generates just a model per iteration and negates its true-valued literals x_a, y_a until the logical model is still meaningful.
In addiction, this technique let the solver mine just rules that are not an expansion of an already discovered one.

Once the full set of rules is extracted, the minimal ones are stored in the output file.

Rule extraction with Apriori

The structure of the program based on Apriori is quite simple:

1. Transactions are extracted from data;
2. Transactions are fed to the `apriori` function;
3. The obtained rules are cleaned and minimized.

This version of ARM will be use as a comparison for the performance of the SAT-based ARM.

Results

In both the version implemented the SAT-based algorithm is not able to return the right set of valid rules (i.e. rules that respect the thresholds).

Y-N The SAT-based finds less valid rules then the Apriori-based. In addiction, it only finds rules if the support threshold is low w.r.t. the transaction database dimension. However, the rules mined are not in contrast with the ones found with Apriori.

Y The SAT-based finds more model then the Apriori-based, g.e. with an high frequency threshold and max confidence ($\alpha = 0.8$, $\beta = 1$) it is able to mine 59 minimal rules. These rules have all $Y =$ `'export_administration_act_south_africa'`, an item with a 80% frequency.

Possible causes

Two factors can affect the behaviour of the SAT-based ARM:

- the Pseudo-Boolean encoding, which makes the number of clauses grow exponentially (hidden variables are added);
- the random picking of literal to check, carried out by the MiniSat solver.

The `PySAT` library allows just basic management of the logical model, so there is no way to influence these two elements of the program.

Table of results

Algorithm	Y-N rules	Y-N time	Y rules	Y time
SAT-based	3	1s	59	40s
Apriori-based	161	25s	0	$<$1s

Table: Results with parameters $\alpha = 0.8$, $|D| =$ *full* for Y ; $\alpha = 0.2$, $|D| = 50$ for Y-N. [β always 1]

- "PySAT Documentation, Release 0.1.4.dev17", Alexey Ignatiev, Joao Marques-Silva, Antonio Morgad, 2019.
- "A SAT-Based Approach for Mining Association Rules", Abdelhamid Boudane, Said Jabbour, Lakhdar Sais, Yakoub Salhi, 2016.
- "Efficient-Apriori Documentation, Release 1.0.0", tommyod, 2019.