CSC 526 Project Report

MARFS

Jennifer Parnell

jp1622@jagmail.southalabama.edu

Introduction

The MARFS algorithm is a data mining algorithm using action rules. MARFS stands for mining action rules from scratch. There is not code available to run this algorithm. Also, two other algorithms can be used in conjunction with MARFS. These are the Apriori algorithm and the FP Growth algorithm. The goal is to find which one is the fastest for the MARFS algorithm.

Proposed Solution and Experimental Setup

The first step is to code the MARFS algorithm with Java using the Apriori algorithm. It will be tested using different data sets. These data sets were provided with the project. They have similar configuration but with different data specifications. For example, some data is numbers and some are words or characters. Success is when the code with the algorithms give results for the each dataset. After a successful implementation, the runtime for the program with the Apriori algorithm will be recorded for each data set. Next, code the FP growth with the MARFS algorithm. After a successful implementation, the runtime for the program with FP growth will be recorded for each data set. The final step is to compare the runtimes for each of the algorithms on the data sets to find the fastest times.

Results

What is the things you saw? What is important about the basic results?

Conclusions and future work

Three main questions (subsection)

1. Did it work? What did and did not work? Was it successful, a failure, in-between? What should people know?
2. What did you learn (tools, techniques, data mining, KDD process, whatever)?
3. Future Work – if someone was to expand this, what should they do?

References (If Any)

He, Zengyou & Xu, Xiaofei & Deng, Shengchun & Ma, Ronghua. (2005). Mining action rules from scratch. Expert Systems with Applications. 29. 691-699. 10.1016/j.eswa.2005.04.031.

Introduction

The ability to mine action rules from data could allow businesses to improve profits from customers. The paper, *Mining action rules from scratch* by He et al., shows an algorithm designed to mine action rules from the data. This paper shows two version of an algorithm known as MARFS1 and MARFS2 [1]. The algorithms are shown in Figure 1 and Figure 2. The purpose of this project is to implement one or both versions of the MARFS algorithm.

Algorithm MARFS1

Input: E = PE ∪ NE

minsupP, minsupN, maxcost, minconf.

Output: Answer // the final set of discovered action rules

L_1 = {frequent and low cost action rules having length *1*};

A_1 = {frequent, low cost and high confidence action rules having length *1*};

for (k =2; $L_{k-1} \neq \emptyset$; k++) **do begin**

C_k= candidates-gen (L_{k-1}); // New candidates

forall example $e^- \in NE$ **do begin**

forall candidates c ∈ C_k **do begin**

if satisfyPL (c, e^-) //the candidate c is not a valid action rule

c.valid =false

if satisfyNL (c, e^-)

c.ncount ++; //negative support

end

forall example $e^+ \in PE$ **do begin**

forall candidates c ∈ C_k **do begin**

if satisfyNL (c, e^+) //the candidate c is not a valid action rule

c.valid =false

if satisfyPL (c, e^+)

c.pcount ++; //positive support

end

L_k = {c ∈ C_k | c.pcount ≥ minsupP and c.ncount ≥ minsupN and c.cost<maxcost}

A_k = conf (L_k) = {c ∈ L_k | c.confidence ≥ minconf and c.valid=true }

end

Answer = $\cup_k A_k$;

Figure 1: MARFS1

Algorithm MARFS2

Input: E = PE ∪ NE

minsupP, minsupN, maxcost, minconf.

Output: Answer // the final set of discovered action rules

1. Find the set of frequent formulas, FN, with frequent pattern algorithm on NE.
2. Find the set of frequent formulas, FP, with frequent pattern algorithm on PE
3. Combining FN and FP to generate candidate action rules
4. Scanning the dataset, test each candidate action rule using threshold parameters and produce answer

Figure 2: MARFS2

The MARFS2 algorithm finds frequent patterns before generating the candidates for the action rules. This will be implemented using the Apriori or FP Growth algorithms to generate the candidates. The evaluation of the runtime for these algorithms determines which one is more efficient for the MARFS algorithm.

Proposed Solution and Experimental Setup

Experimental Methodology

This project creates a version of the MARFS algorithm to use on datasets. Then, it will evaluate the results of the runtime on different algorithms in conjunction with the MARFS. The language is Java in the eclipse IDE. The Apriori and FP Growth algorithms are provided by Free Software Foundation, Inc. The data sets were provided with the project details, where they have similar format but varying data types.

The first step is to code the MARFS2 algorithm with Java using the Apriori algorithm. It will be tested using different data sets. These data sets were provided with the project. They have similar configuration but with different data specifications. For example, some data is numbers, and some are words or characters. Two samples of the data sets are shown in Figures 3 and 4. The first step in the code is to format the data for the Apriori and FP Growth algorithms. These algorithms need the data to have unique numbers for the attribute data. While formatting the data the code also sorts it into two categories of positive and negative groups. The positive group is based on a user selected attribute data option. Then the negative group is all data that is not positive. These groups are saved into their own separate data files.

```
@class balance:L,B,R
@flexible Left_Weight:1,2,3,4,5
@flexible Left_Distance:1,2,3,4,5
@flexible Right_Weight:1,2,3,4,5
@flexible Right_Distance:1,2,3,4,5
@data
B,1,1,1,1
R,1,1,1,2
R,1,1,1,3
R,1,1,1,4
R,1,1,1,5
R,1,1,2,1
R,1,1,2,2
```

Figure 3: balance.

```
@stable buying:v-high,high,med,low
@stable maint:v-high,high,med,low
@flexible doors:2,3,4,5-more
@stable persons:2,4,more
@flexible lug_boot:small,med,big
@flexible safety:low,med,high
@class acceptability:unacc,acc,good,vgood
@data
vhigh,vhigh,2,2,small,low,unacc
vhigh,vhigh,2,2,small,med,unacc
vhigh,vhigh,2,2,small,high,unacc
vhigh,vhigh,2,2,med,low,unacc
vhigh,vhigh,2,2,med,med,unacc
vhigh,vhigh,2,2,med,high,unacc
vhigh,vhigh,2,2,big,low,unacc
```

Figure 4: car.data

The next step for the code is to implement the Apriori and FP Growth algorithms. Once the data is formatted correctly and in separate data files, the program uses the algorithms. The algorithms have built in timers for the runtime. The algorithms need a user defined minimum support number for both positive and negative groups. After implementing the algorithms, the results show the candidate action rules and times for runtime.

The final steps for the MARFS2 algorithm are to compare the candidates and calculate the minimum confidence. The comparison of the candidates eliminates all candidates that are found in the opposite group. Then the confidence is calculated and compared to the user defined minimum confidence. All candidates that are not in the opposite group and meet the confidence are given as results. The candidate results are logged with the runtimes for the Apriori and FP Growth algorithms.

Results

The results are not complete at this time. The MARSF2 algorithm code is almost complete but is missing the final steps of eliminating candidates that belong to the opposite group and the minimum confidence calculation. There are many reasons for the incomplete project. The first reason was issues with getting the data sets, code, and other parts of the project. The next problem arose with the formatting of the data. It took some time to get it to create unique numbers based on all the different data items because of the different data types. This also had to account for the data transforming with the correct unique number throughout the data set based on the attribute. The final issue is time left to figure out the usage of the custom class of Itemset to compare the candidates. All issues have been fixed and the algorithm is working correctly.

The results for the runtimes for the Apriori and FP Growth algorithms on each data set. They are shown in Table 1. There were problems with the FP Growth algorithm when using the larger data sets. From Table 1, the FP Growth algorithm is faster than the Apriori algorithm.

		Apriori Time (ms)			FP Growth Time (ms)		
	File length	Positive	Negative	Total	Positive	Negative	Total
Balance	625	12	7	19	4	1	5
Car	1728	24	4	28	10	3	13
Hepatitis	155	55	8	63	13	2	15
Nursery	12960	7	78	85	3	51	54
Tictac	958	27	8	35	10	4	14

Table 1

Conclusions and future work

This project showed the most efficient algorithm to use in conjunction with the MARFS algorithm.

What did you learn

The importance of documenting and explaining algorithms and methods when writing papers. When there are gaps of knowledge or documentation, it does not allow others to use the algorithm or methods in the future.

Future Work

Implement the MARFS1 algorithm.

References ()

[1] He, Zengyou & Xu, Xiaofei & Deng, Shengchun & Ma, Ronghua. (2005). Mining action rules from scratch. Expert Systems with Applications. 29. 691-699. 10.1016/j.eswa.2005.04.031.