AN IMPLEMENTATION FOR A HEURISTIC RANGE-BASED CLASSIFICATION METHOD

PHUONG ANH TRAN

MSc Computing

A DISSERTATION SUBMITTED FOR THE DEGREE OF MSC COMPUTING

DEPARTMENT OF COMPUTER SCIENCE

CARDIFF UNIVERSITY

2016

Declaration

I hereby declare that this dissertation is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the dissertation.

This dissertation has also not been submitted for any degree in any university previously.

Phuong Anh Tran

June 2016

Acknowledgments

To my supervisor, Dr Jinhua Shao, whom inspires me with confidence and beyond.

Abstract

Contents

List of Figures

List of Tables

List of Algorithms

Section 1

Introduction

The ability to extract meaningful information from a set of data has been a crucial problem within data mining and machine learning research. One approach is to recognise certain patterns within a dataset, then transform these patterns into rules. In the future, these rules could be used to predict an classified outcome for any given instance of that dataset. The ability to transform such patterns is thus a deciding factor in producing accurate predictions. Subsequently, its accuracy relies on the choices of methods and algorithms in learning and deriving useful and relevant rules.

The goal of this project is to investigate an algorithm to derive classification rules for range-based data. Ranges pose a more challenging problem to assess since there are more possible patterns to deduce than that of categorical values. Therefore, before patterns can be learnt, the data must be split into 'right intervals' [src] so that the rules devised will be more fitting. The process of deducing relevant subset of data thus plays a significant part in deciding the rule accuracy, and an important step before actual classification can take place.

In details, this project analyse and implement a method that aims to solve the above issues i.e. to classification rules for range-based numerical dataset, developed in a previously published paper by Shao, J. and Tziatzios, A. [src]. The original method withholds two distinctive features: firstly, the data trimming process is done by an algorithm similar to Kadane's [src] solving maximum-array problem, which is highly efficient due to its linear time complexity (logN). Secondly, in order to mimic that max-sum method, chosen ranges are analysed by

assigning binary tags upon their covered items depending on the items' class values. By using class values, the given ranges can be narrowed down to to be more relevant to the rules derived. These two features make for a fast and efficient way to classify range-based dataset. This method is then analysed further in order to produce an improved[?] version that is implemented as a final software using Java.

[What was improved?] [Outcome? Your own work?]

Section 2

Background

The project implements an algorithm originated from [SRC]. The algorithm utilises a similar approach to association rule mining: it searches for associated ranges in each attribute, then combine these ranges to form a larger rule. To make this search more efficient, class values are used to guide the search i.e. only ranges that have relevant class tags are considered. The algorithm is explained in the following sections.

2.1 Preliminaries

Assuming that we have a dataset that could be presented as a table $T(A_1, A_2, A_3, ..A_m, C)$, where $A_{j,m}$ as $1 \leq j \leq m$ are range attributes and C is the categorical class value. A single tuple within T is denoted as $t_k = (v_1, v_2, v_3..v_m, C)$ where v_m is a value under A_m.

A sample dataset served as an example for this section can be shown in Table 2.1 below:

Definition 1 (range) Assume within the domain of attribute A exists two values a and b that represents a continuous range over A_j, denoted $r = [a, b]_{A_j}$. This range covers a set of values in A that lies between a and b.

Example 1: From Table 2.1, a range of $[0.64, 0.75]_{A_1}$ would cover a set of values 0.64, 0.71 and 0.75 in A_1.

Definition 2 (cover) Assume $r = [a, b]_{A_j}$ to be a range over attribute A. This range r covers a set of tuples where their values are between a and b where $a \leq v_j \leq b$. This set of tuples that is covered by r is denoted $\tau(r)$.

Table 2.1: A sample dataset as table

T	A_1	A_2	A_3	A_4	C
t1	0.75	1.45	2.13	4.56	c_1
t2	0.64	1.62	2.64	4.75	c_2
t3	0.71	1.21	3.11	3.97	c_1
t4	0.57	1.23	2.75	4.24	c_1
t5	0.80	1.53	2.34	4.11	c_2

Example 2: From Example 1, a range of $[0.64, 0.75]_{A_1}$ would cover a set of tuples t_2, t_3, t_1 hence its cover is t_2, t_3, t_1.

Definition 3 (associated ranges) Assume $r_1 = [a_1, b_1]_{A_1}$ to be a range over A_1 and $r_2 = [a_2, b_2]_{A_2}$ to be a range over A_2. Those ranges are associated ranges if $\tau(r_1) \cap \tau(r_2) \neq \varnothing$

Example 3: Assume $r_1 = [0.64, 0.75]_{A_1}$ and $r_2 = [1.21, 1.45]_{A_2}$. Table 2.1 shows that r_1 covers t_1, t_2, t_3 and r_2 covers t_1, t_4, t_3, which means $r_1 \cap r_2 = \{t_1, t_2, t_3\} \cap \{t_1, t_3, t_4\} = \{t_1, t_3\}$. Since there are mutual tuples between these ranges, r_1 and r_2 are associated ranges.

Definition 4 (range-based classification rule) Assume c to be a class value from C and $r_1, r_2, r_3..r_h$ be a set of associated ranges. $r_1, r_2, r_3..r_h \rightarrow c$ is a range-based classification rule.

By definition, any tuple t_k can form a rule. However, this will not be accurate nor useful. To determine whether a rule is qualified, there are criteria to be used, which are to be discussed in the sub-section below.

2.2 Criteria

There are three criteria used in this algorithm: support, confidence and density. This sub-section presents the formal definitions for those measures, while examples are to be

provided in the next sub-section where the actual algorithm is reviewed.

Definition 5 (support) Assume T to be a table and $r_1, r_2, r_3..r_h \rightarrow c$ to be a range-based classification rule derived from T. The support for r, provided $\mid . \mid$ being the size of a set, is:

$$\sigma(r) = \frac{\mid \tau(r_1) \cap \tau(r_2) \cap ... \cap \tau(r_h) \mid}{\mid T \mid}$$

Definition 6 (confidence) Assume similar settings from Definition 5, the confidence for r in T is:

$$\delta(r) = \frac{\mid \tau(r_1) \cap \tau(r_2) \cap ... \cap \tau(r_h) \cap \tau(c) \mid}{\mid \tau(r_1) \cap \tau(r_2) \cap ... \cap \tau(r_h) \mid}$$

Definition 7 (density) Assume similar settings from Definition 5, the density for r in T is:

$$\gamma(r) = \frac{\mid \tau(r_1) \cap \tau(r_2) \cap ... \cap \tau(r_h) \cap \tau(c) \mid}{\mid \tau(r_1) \cup \tau(r_2) \cup ... \cup \tau(r_h) \mid}$$

2.3 Original algorithm

As mentioned, the algorithm provides a more efficient way to discover associated ranges than the traditional brute-force solution. Instead of traversing through all possible combinations of ranges among all attributes, this algorithm first divides the data into sub-data pools in accordance to its class value. Then, under each attribute, it searches for sub-ranges that could pass certain user-defined thresholds (confidence, support and density). These sub-ranges are then collected and ready to be tested for possible combinations between each other that passes certain thresholds similar to the previous step, which are then collected and ready to be tested for larger combinations. Additionally, after larger combinations are found, an extra adjustment step is done on class values of relevant tuples within the new associated ranges. Afterwards, new associated ranges are ready to be analysed again to find more possible sub-ranges and new combination of sub-ranges. This iteration continues until no possible combinations can be found, so that the last standing associated ranges are translated into the concluding classification rule for that class value.

To get a clearer structure of the algorithm, its proceedings can be divided into two phases:

Phase I (Analyse) This phase has its primary concern as to analyse ranges. Based on the algorithm description, it includes:

1. Partition data into sub-data pool using class values.
2. For each attribute in sub-data pool, find a maximum and minimum range within that attribute and use it as a starting range for next step.
3. From min-max starting ranges, search for sub-ranges that passes certain user-specified thresholds such as confidence, support and density. Sub-ranges are collected and passed onto Phase II to generate larger associated ranges.

Phase II (Generate) This phase has its primary concern as to generate larger associated ranges for the next iteration. Based on the algorithm, it includes:

1. For each sub-range obtained from last phase, check for possible combination of that range with the remaining ones.
2. Adjust class values of relevant tuples$r_1, r_2, r_3..r_h \rightarrow c$ under the new combined sub-range.

Details for each phase are as follows:

2.3.1 Phase I (Analyse)

Given a table $T(A_1, A_2, A_3, ..A_m, C)$, the algorithm attemps to seek for classification $r_1, r_2, r_3..r_h \rightarrow c_j$ by finding, among instances that are under class $c_j \subset C$, a number of associated ranges $r_1, r_2, r_3..r_h$ that have at least a minimum support (σ_{min}), density (θ_{min}) and confidence (γ_{min}) that had previously been specified by the user. This process can be explained in Phase I (Analyse) in which the algorithm seeks for associated ranges and checks if thresholds are satisfied. In pseudo-code, this phase is outlined as below: