Temporal Event Sequence Rule Mining

Sherri K. Harms
University of Nebraska at Kearney, USA

INTRODUCTION

The emergence of remote sensing, scientific simulation and other survey technologies has dramatically enhanced our capabilities to collect temporal data. However, the explosive growth in data makes the management, analysis, and use of data both difficult and expensive. Methods that characterize interesting or unusual patterns from the volumes of temporal data are needed (Roddick & Spiliopoulou, 2002; Han & Kamber, 2005).

The association rule mining methods described in this chapter provide the ability to find periodic occurrences of inter-sequential factors of interest, from groups of long, non-transactional temporal event sequences. Association rule mining is well-known to work well for problems related to the recognition of frequent patterns of data (Han & Kamber, 2005). Rules are relatively easy for humans to interpret and have a long history of use in artificial intelligence for representing knowledge learned from data.

BACKGROUND

A time series database contains sequences of values typically measured at equal time intervals. There are two main categories of temporal sequences: *transaction-based sequences* and *event sequences*. A transaction-based sequence includes an identifier such as a customer ID, and data mining revolves around finding patterns within transactions that have matching identifiers. An example pattern is "A customer who bought Intel stock is likely to buy Google stock later." The transaction has a definite boundary around known items of interest. There are many techniques that address these problems (Han & Kamber, 2005).

Data analysis on event sequences is enormously more complex than transactional data analysis. Event sequences are often long streams of data where interesting patterns occur either within the sequence or across multiple sequences. There are no inherently defined boundaries (or identifiers) around factors that might be of interest. Temporal event sequence algorithms must be able to compute inference from volumes of data, find the interesting events involved, and define the boundaries around them. An example pattern is "A La Niña weather pattern is likely to precede drought in the western United States". La Niña weather data is based on Pacific Ocean surface temperatures and atmospheric values, and drought data is based on precipitation data from weather stations throughout the western United States. The sheer number of possible combinations of interesting factors and relationships between them can easily overwhelm human analytical abilities. Often there is a delay between the occurrence of an event and its influence on dependent variables. These factors make finding interesting patterns difficult.

Many different methods have been applied to temporal event sequences. In statistics, event sequence data is often called a marked point process. However, traditional methods for analyzing marked point processes are ill suited for problems with long, non-transactional sequences with numerous event types (Mannila et al. 1997). The association rule mining methods described in this chapter extract meaningful inter-sequential patterns in this type of data. Additionally, the mined rules provide much richer information than correlation coefficients from correlating entire sequences. The methods described in this chapter are similar to inductive logic programming, but with an emphasis on time-limited occurrences of sequential data. Similarities also exist to algorithms used in string matching and bioinformatics, but the classes of patterns differ (Mannila et al. 1997).

MAIN FOCUS

Mining association rules is typically decomposed into three sub-problems: 1) prepare the data for analysis, 2) find frequent patterns and 3) generate association rules from the sets representing those frequent patterns (Agrawal et al., 1993).

Preparing Event Sequences for Analysis

To prepare sequential data for association rule mining, the data is discretized and partitioned into sequences of events. Typically, the data is normalized and segmented into partitions that have similar characteristics within a given interval. Each partition identifier is called an event type. Different partitioning methods and interval sizes produce diverse discretized versions of the same dataset. Proper discretization relies on domain-expert involvement. When multivariate sequences are used, each is normalized and discretized independently.

Partitioning methods include symbolizing (Lin et al., 2003) and intervals (Hoppner, 2002). The assignment of values to a certain state is somewhat arbitrary near the decision boundaries. Mörchen & Ultsch (2005) presented a method for meaningful unsupervised discretization that reduces the vulnerability to outliers in the data and reduces the problems that occur when intervals are cut in high density regions of data values.

Mielikäinen et al. (2006) proposed a discretization technique that segments data by aggregating the results of several segmentation algorithms and choosing the discretization that agrees as much as possible with the underlying structure of the data.

Finding Frequent Episodes based on Sliding Window Technologies

A discretized version of the time series is referred to as an *event sequence*. An event sequence $\hat{S}$ is a finite, time-ordered sequence of events (Mannila et al., 1995). That is, $\hat{S} = (e_1, e_2, \ldots e_n)$. An event is an occurrence of an event type at a given timestamp. The time that a given event e_i occurs is denoted i, and $i \leq i+1$ for all i timestamps in the event sequence. A sequence includes events from a single finite set of event types. An event type can be repeated multiple times in a sequence. For example, the event sequence $\hat{S}_1 = AABCAB$ is a sequence of 6 events, from a set of 3 event types {A,B,C}. In this event sequence, an A event occurs at time 1, followed by another A event, followed by a B event, and so on. The step size between events is constant for a given sequence.

An *episode* in an event sequence is a combination of events with partially specified order (Mannila et al., 1997). It occurs in a sequence if there are occurrences of events in an order consistent with the given order, within a given time bound. Formally, an episode α is a pair *(V, ordering),* where V is a collection of events and the ordering is *parallel* if no order is specified, and *serial* if the events of the episode have fixed order. The episode length is defined as the number of events in the episode.

Finding frequent episodes in sequences was first described by Mannila et al. (1995). Frequent episodes are discovered by using a sliding window approach, *WINEPI*. A *window* on an event sequence $\hat{S}$ is an event subsequence, $w = e_i, e_{i+1}, \ldots e_{i+d}$ where the width of window w, denoted d, is the time interval of interest. The set of all windows w on $\hat{S}$, with a width of d is denoted $\hat{W}(\hat{S},d)$. In this system, the value of the window width is user-specified, varying the closeness of event occurrences. To process data, the algorithm sequentially slides the window of width d one step at a time through the data. The *frequency* of an episode α is defined as the fraction of windows in which the episode occurs. For example, in the sequence $\hat{S}_1$ above, if a sliding window of width 3 is used, serial episode α = AB occurs in the first window (AAB), the second window (ABC), and the fourth window (CAB). The guiding principle of the algorithm lies in the "downward-closed" property of frequency, which means every subepisode is at least as frequent as its superepisode. Candidate episodes with $(k+1)$ events are generated by joining frequent episodes that have k events in common, and episodes that do not meet a user-specified frequency threshold are pruned.

The closure principle was first applied to the event sequences by Harms et al. (2001) to use only a subset of frequent episodes, called frequent closed episodes. A closed sequential pattern is a sequential pattern which has no super-sequence with the same occurrence frequency. A frequent closed episode is the intersection of all frequent episodes containing it. For example, in the $\hat{S}1$ sequence, using a window width d = 3, and a minimum frequency of three windows, serial episode α = AB is a frequent closed episode since no larger frequent episode contains it, and it meets the minimum frequency threshold. Using closed episodes results in a reduced input size and in a faster generation of the episodal association rules, especially when events occur in clusters. Casas-Garriga (2005) added post-processing of closed sequences to generate classical partial orders, without dealing directly with input data.

Hoppner & Klawonn (2002) divided multivariate sequences into small segments and discretized them based on their qualitative descriptions (such as increas-