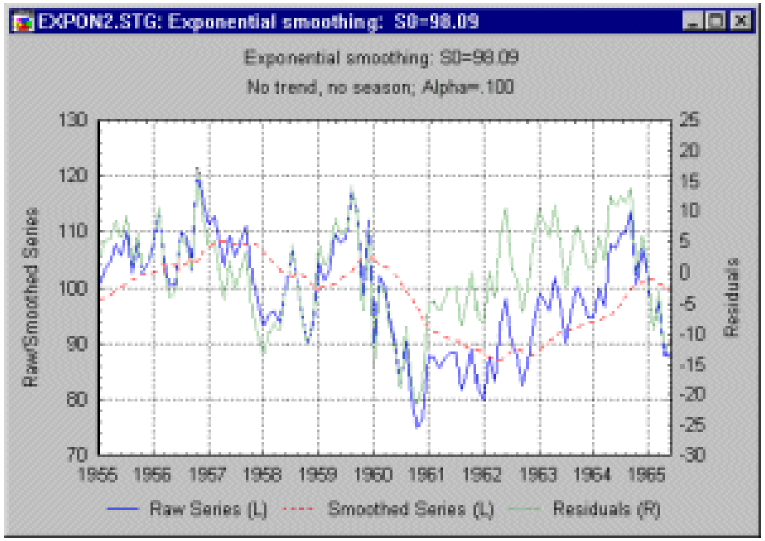
“Temporal Data Mining”



Outline

- Motivation for Temporal Data Mining (TDM)
- Examples of Temporal Data
- TDM Concepts
- Sequence Mining: temporal association mining
- Calendric Association Rules
- Trend Dependencies
- Frequent Episodes
- Markov Models & Hidden Markov Models

Motivation for Temporal Data Mining: Time-varying versus Space-varying Processes

- Most of the data mining techniques that we have discussed so far have focused on the classification, prediction, or characterization of single data points.
- For example:
  - "Assign a record to one of a set of classes" --- techniques :
    - Decision Trees; Neural Networks; Bayesian classifiers; etc.
  - "Predict the value of a field in a record given the values of the other fields" --- techniques :
    - Regression; Neural Networks; Association rules; etc.
  - "Find regions of feature space where data points are densely grouped" --- techniques :
    - Clustering; Self-Organizing Maps

Time-varying versus Space-varying Processes: The Statistical Independence Assumption

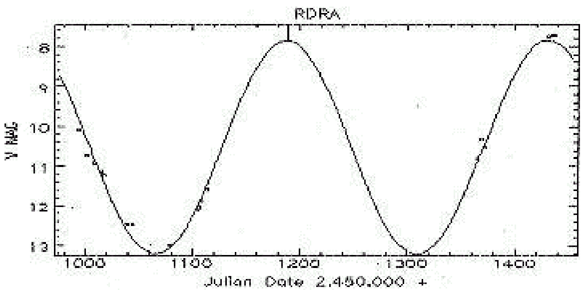
- In the methods that we have considered so far, we have assumed that each observed data point is **statistically independent** from the observation that preceded it. For example:
  - Classification: the class of data point $\mathbf{x}_k$ is not influenced by the class of $\mathbf{x}_{k-1}$ (or indeed by any other data point).
  - Prediction: the value of a specific field in a given record depends only on the values of the fields within that record, not on values in the fields of any other records.
- Many important real-world data mining problems do **not** satisfy this Statistical Independence Assumption.

Motivation for Temporal Data Mining, continued

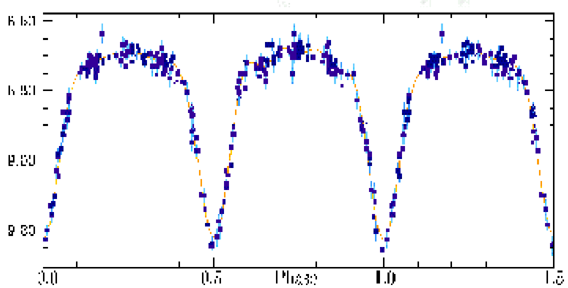
- There are many examples of time-ordered data (e.g, scientific, medical, financial, sports, computer network traffic, web logs, sales transactions, factory machinery performance, fleet repair records, weather/climate, stock market, telephone calls, etc.)
- Time-tagged data collections frequently require repeated measurements -- dynamic data volumes are therefore growing, and continuing to grow!
- Many of the data mining methods that we have studied so far require some modification to handle temporal relationships ("before", "after", "during", "in summer", "whenever X happens")
- Time-ordered data lend themselves to prediction -- what is the likelihood of an event, given the preceding history of events (e.g, failure of parts in an electro-mechanical system)?
- Time-ordered data often link certain events to specific patterns of temporal behavior (e.g, network intrusion break-ins).

Temporal Data Examples

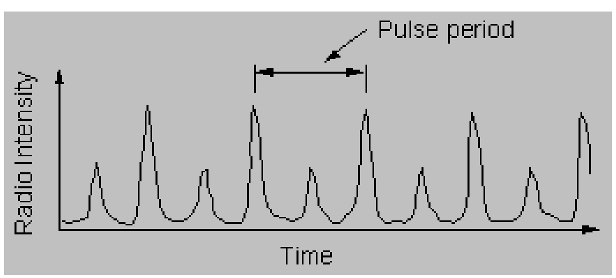
- Periodic -- sinusoidal:



- n-sine:

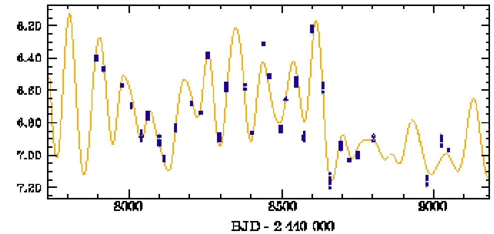


- Periodic -- spiked events:

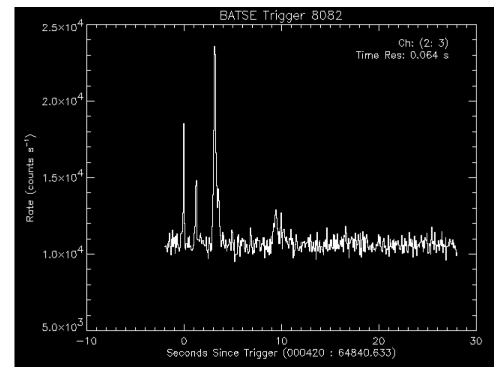


(Chirp)

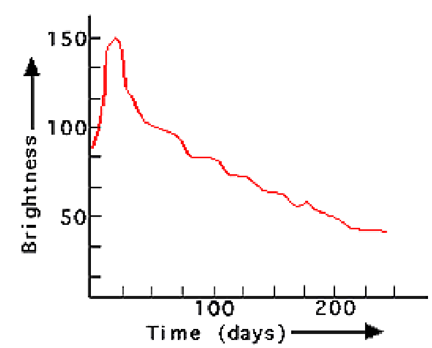
- Aperiodic events (noise?):



- Single spiked events:



- Single long-duration events:



Trivial Temporal Data Examples

- A flat line (constant measurement of some quantity).
- A linearly increasing or decreasing curve (straight line).

- Even though these are not complex, they are still temporal data. Fortunately, these data streams can easily be represented by one or two parameters, and then you're done!

Temporal Data Mining (TDM) Concepts

- Event: the occurrence of some data pattern in time
- Time Series: a sequence of data over a period of time
- Temporal Pattern: the structure of the time series, perhaps represented as a vector in a Q-dimensional metric space, used to characterize and/or predict events
- Temporal Pattern Cluster: the set of all vectors within some specified similarity distance of a temporal pattern
- Phase Space: a state space of metrics that describe the temporal pattern (e.g., Fourier space, wavelets, …)
- Event Characterization Function: connects events to temporal patterns; characterizes the event in phase space

TDM Concepts, continued

- Absolute Time Cycle Events: Co-occurrence of two or more events at the same time
- Contiguous Time Cycle Events: Co-occurrence of two or more events in consecutive time intervals

Sequence Mining

- Temporal Association Mining is Sequence Mining.
- Temporal data need to be "categorized" (e.g, by season, or month, or day of week, or time interval, such as morning, afternoon, night).
- The number of pair-wise associations could become huge (= combinatorial explosion), and so care must be taken in selecting categorizations.
- Sequence mining must include concepts of "*before*" and "*after*". And it can include *time intervals* for the "*before*" and "*after*" items. (e.g., it is found that people who have purchased a VCR are three times more likely to purchase a camcorder within the next two to four months.)

Calendric Association Rules

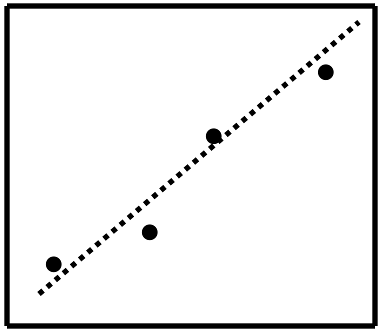
- Each data item d_i in a temporal database D is associated with a timestamp t_i.
- The database time range is assumed to be divided into predefined time intervals, each having a duration t. Interval k is defined by:

$$\text{Interval } k\text{: } kt < (t_i - t_0) < (k + 1)t.$$

- The k^{th} subset of D is $D[k]$, corresponding to the set of all data items with timestamps in interval k.
- Partitional Association Rule Mining is applied to each partition $D[k]$ separately, to search for associations with high confidence that occur within specific time intervals k. These are **Calendric Association Rules**.

Trend Dependencies

- Initial thought (this is not serious):
  - Two points define a tendency
  - Three points define a trend
  - Four points define a theory



- Seriously… Trend Dependencies are association rules that discover time variations in attribute values (e.g, an employee's salary always increases over time).
- More generally, Trend Dependencies do not have to be time-dependent variations. Trends can occur in any kind of data, not just time-based data.

Frequent Episodes

- Frequent Episodes = a set of events that occur within a defined time-span; for example, a recording of all computer network connections that a single user made within five minutes.
- An Episode Rule is a generalized association rule applied to sequences of events.
- An Event Sequence is an ordered list of events, each occurring at a particular time.
- Episode rules are used to predict failures in communications equipment (switching nodes): data mining is used to find a predictive model to predict when an event (failure) will occur based upon a sequence of earlier events (= a **Markov Chain**).

Outline

- Motivation for Temporal Data Mining (TDM)
- Examples of Temporal Data
- TDM Concepts
- Sequence Mining: temporal association mining
- Calendric Association Rules
- Trend Dependencies
- Frequent Episodes
- Markov Models & Hidden Markov Models

Dependent Sequences - Markov Chains

- We often encounter **sequences** of observations in which each observation may depend on the observations which preceded it. This is a Markov Chain (sometimes spelled "Markoff"). If the observation depends only on the immediately preceding value, and no other, then this is called a First-Order Markov Chain.
- Examples
  - Sequences of phonemes (fundamental sounds) in speech -- used in speech recognition.
  - Sequences of letters or words in text -- used in text categorization, information retrieval, text mining.
  - Sequences of web page accesses -- used in web usage mining.
  - Sequences of bases in DNA -- used in genome mining.
  - Sequences of pen-strokes -- used in hand-writing analysis.
- In all these cases, the probability of observing a particular value in the sequence can depend on the values which came before it.

Sequence Example: Web Log

- Consider the following extract from a web log:

xxx - - [16/Sep/2002:14:50:34 +1000]	"GET /courseware/cse5230/ HTTP/1.1"	200	13539
xxx - - [16/Sep/2002:14:50:42 +1000]	"GET /courseware/cse5230/html/research_paper.html HTTP/1.1"	200	11118
xxx - - [16/Sep/2002:14:51:28 +1000]	"GET /courseware/cse5230/html/tutorials.html HTTP/1.1"	200	7750
xxx - - [16/Sep/2002:14:51:30 +1000]	"GET /courseware/cse5230/assets/images/citation.pdf HTTP/1.1"	200	32768
xxx - - [16/Sep/2002:14:51:31 +1000]	"GET /courseware/cse5230/assets/images/citation.pdf HTTP/1.1"	206	146390
xxx - - [16/Sep/2002:14:51:40 +1000]	"GET /courseware/cse5230/assets/images/clustering.pdf HTTP/1.1"	200	17100
xxx - - [16/Sep/2002:14:51:40 +1000]	"GET /courseware/cse5230/assets/images/clustering.pdf HTTP/1.1"	206	14520
xxx - - [16/Sep/2002:14:51:56 +1000]	"GET /courseware/cse5230/assets/images/NeuralNetworksTute.pdf HTTP/1.1"	200	17137
xxx - - [16/Sep/2002:14:51:56 +1000]	"GET /courseware/cse5230/assets/images/NeuralNetworksTute.pdf HTTP/1.1"	206	16017
xxx - - [16/Sep/2002:14:52:03 +1000]	"GET /courseware/cse5230/html/lectures.html HTTP/1.1"	200	9608
xxx - - [16/Sep/2002:14:52:05 +1000]	"GET /courseware/cse5230/assets/images/week03.ppt HTTP/1.1"	200	121856
xxx - - [16/Sep/2002:14:52:24 +1000]	"GET /courseware/cse5230/assets/images/week06.ppt HTTP/1.1"	200	527872

- Clearly the URL that is requested depends on the URL that was requested before
  - If the user uses the “Back” button in his/her browser, the requested URL may depend on earlier URLs in the sequence too.
- Given a particular observed URL, we can then calculate the **probabilities** of observing other possible URLs in the next click.
  - Note that we may even observe the **same** URL next.

First-Order Markov Models

- In order to model processes such as these, we make use of the idea of **states**. At any time *t*, we consider the system to be in state *w(t)*.
- We can consider a sequence of successive states -- this sequence has length *T* :

$$\mathbf{w}_T = \{w(1), w(2), \ldots, w(T)\}$$

- We can model the production of a particular sequence using *transition probabilities*:

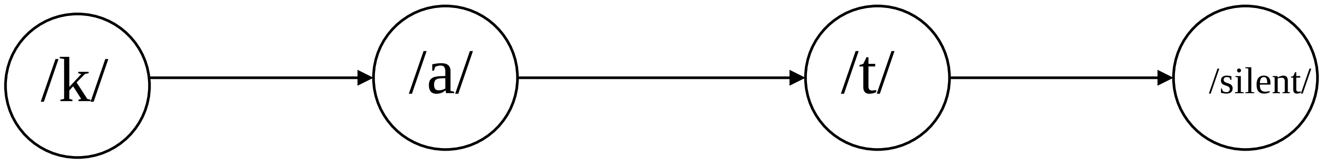
$$P(w_j(t+1) \mid w_i(t)) = a_{ij}$$

  – This a_{ij} is the probability that the system will be in state w_j at time $t+1$ given that it was in state w_i at time t.

First-Order Markov Models, continued

- A model of states and transition probabilities, such as the one we have just described, is called a **Markov Model**.
- Since we have assumed that the transition probabilities depend only on the one immediately preceding state, this is a **First-order Markov Model.**
  - Higher-order Markov models are possible (dependent on multiple previous states), but we will not consider them here.
- For example, Markov models for human speech could have states corresponding to the various phonemes:
  - A Markov model for the word "cat" would have states for /k/, /a/, /t/ and a final silent state.

Example: Markov Model for "cat"



- /k/, /a/, and /t/ correspond to the phonemes (sounds of speech) that lead to the pronunciation of the word "cat".
- The probability that /a/ will follow /k/ is based upon a "dictionary" of speech usage: the probability that /a/ will follow /k/ is the frequency of occurrence of that "state transition" in the dictionary. Likewise, for the transition from /a/ to /t/, and from /t/ to the /silent/ state, the probability is calculated from the frequency of occurrence of that state transition in the dictionary.
- These probabilities can be calculated (i.e., they are observed).
- The "dictionary" may be the existing database (=training set).

So, what does this have to do with anything? (i.e., what does it have to do with Data Mining?)

- One type of data mining is “Predictive”.
- Temporal data mining is often predictive.
- What are we predicting? -- usually, one wants to predict what will happen next, or what is the probability that a certain thing (e.g, an error condition or failure state or medical condition) will happen.
- How does one make the prediction? -- by using prior frequencies of state transitions, based upon the existing (historical) sequences of data items (state transitions) in the temporal database.
- Thus, Markov Models can be applied to this form of **predictive data mining**. This is **another example of Bayes classification** = probabilistic estimation based upon prior probabilities.

Hidden Markov Models (HMM)

- In the preceding "cat" example, we said that the states correspond to phonemes (the fundamental sounds of speech).
- In a speech-recognition system, however, we don't have access to phonemes – we can only measure properties of the sound produced by a particular speaker (= the training set).
- In general, our observed data do not correspond directly to an underlying state of the model --- The data correspond only to the **visible states** of the system.
  - The visible states are directly accessible for measurement.
- The system can also have internal **"hidden" states**, which cannot be observed directly.
  - For each hidden state, there is a probability of observing each visible state.
- This sort of model is called a **Hidden Markov Model (HMM).**

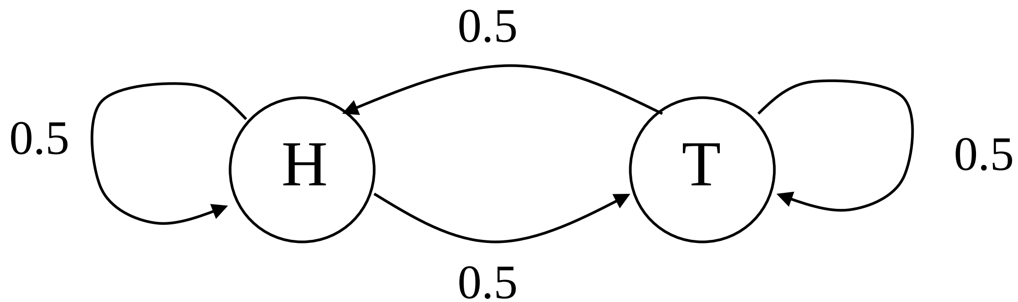
Example: Coin Toss Experiments

- Let us imagine a scenario where we are in a room which is divided in two by a curtain.
- We are on one side of the curtain, and on the other side is an unseen person who will carry out a procedure using coins resulting in a head (H) or a tail (T).
- When the person has carried out the procedure, they call out the result, H or T, which we record.
- This system will allow us to generate a sequence of H's and T's, such as:

```
HHTHTHTTHTTTTTHHTHHHHTHHHTTHHHHHHTTTT
TTTTHTHHTHTTTTTHHTHTHHHTHTHHTTTTHHTTT
HHTHHTTTHTHTHTHTHHHTHHTTHT....
```

Example: A Single Fair Coin

- Imagine that the person behind the curtain has a single fair coin (i.e., it has equal probabilities of coming up heads H or tails T).
- We could model the process that produces the sequence of H's and T's as a Markov model: with two states, and with equal transition probabilities:



- Note that here the visible states correspond exactly to the internal states – the model is not hidden.
- Note also that states can transition to themselves. In other words, if a Head is tossed on one turn, then there is still a 50% chance that a Head will be tossed on the next turn.

Modified Example: A Fair and a Biased Coin

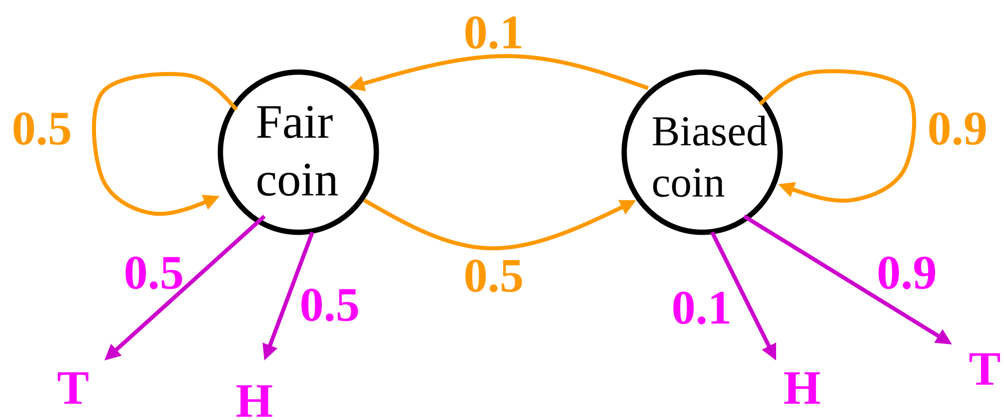
- Now imagine a more complicated scenario. The unseen person behind the curtain has two coins, one fair and one biased [for example, P(T) = 0.9 = 90% in favor of tossing a Tail].
  - (1) The person starts by picking a coin a random.
  - (2) The person tosses the coin, and calls out the result (H or T).
  - (3) If at any time the result is H, the person switches coins.
  - (4) Go back to step (2), and repeat.
- This modified process generates sequences like the following :

```
TTTTTTTTTTTTTTTTTTTTTTTTHHTTTTTTTHHTTTTTTT
TTTTTTTTTTTTTTTHHTTTTTTTTTHTTHTTHHTTTTTHHT
TTTTTTTTTHHTTTTTTTTHTHHHTTTTTTTTTTTTTTHHTT
TTTTTHTHTTTTTTTHHTTTTT...
```

- Note that this looks quite different from the sequence of tosses generated in the "fair coin" example (see Lecture Slide #23).

A Fair and a Biased Coin, continued

- In this scenario, the visible state no longer corresponds exactly to the hidden state of the system:
  - **Visible state: output of H or T (this is what we can see)**
  - **Hidden state: which coin was tossed (we cannot know)**
- We can model this 2-coin process using a HMM:



A Fair and a Biased Coin, continued

- We see from the diagram on the preceding slide that we have extended our model:
  - The visible states are shown in purple, and the *emission probabilities* (probabilities of tossing H or T) are shown too.
- With the internal states *w(t)* and state transition probabilities a_{ij}, we also have visible states *v(t)* and emission probabilities b_{jk} .

$$P(v_k(t) \mid w_j(t)) = b_{jk}$$

  - This is the probability that we will observe a specific visible state, given a particular internal (hidden) state.
  - Note that the b_{jk} do not need to be related to the a_{ij} [they are *coincidentally* the same in the "fair coin and biased coin" example above -- as a result of step (3) on Lecture Slide #25].
- We now have a full Hidden Markov Model (HMM).

HMM: formal definition

- We can now give a more formal definition of a First-order Hidden Markov Model (adapted from [RaJ1986]) :
  - There is a finite number of (internal) states, *N*.
  - At each time *t*, a new state is entered, based upon a transition probability distribution which depends on the state at time $t - 1$. Self-transitions are allowed.
  - After each transition is made, a symbol is output, according to a probability distribution which depends only on the current state. There are thus *N* such probability distributions.
- Estimating the number of states N, as well as the transition and emission probabilities for each state, is usually a very complex problem, but solutions do exist.
- Reference: [RaJ1986] L. R. Rabiner and B. H. Juang, *An introduction to hidden Markov models*, IEEE Magazine on Acoustics, Speech and Signal Processing, 3, 1, pp. 4-16, January 1986.

Use of HMMs

- We have now seen what sorts of processes can be modeled using HMMs, and how an HMM is specified mathematically.
- We now consider how HMMs are actually used.
- Consider the two H and T sequences we saw in the previous examples:
  - How could we decide which coin-toss system was most likely to have produced each sequence?
- To which system would you assign these sequences?

```
1: TTTHHTTTTTTTTTTTTTHHTTTTTTHHTTTHH
2: THHTTTHHHTTHTHTTHTHHTTHHHTTHTHTHT
3: THHTHTHTHTHHHTTHTTTHHTTHTTTTTHHHT
4: HTTTHTTHTTTTHTTTHHTTHTHTTTTTTTTHT
```

- We could answer this question using a Bayesian formulation.