Data Mining

May 24, 2002 Michael Minock

Data Mining and Knowledge Discovery in Databases (KDD)

'Data mining' is like saying 'dirt mining'. But the name stuck.

Loosely speaking techniques come from *statistics* and *machine learning*.

KDD (Knowledge Discovery in Databases) is an overall process of which Data Mining is a component:

KDD: Selection, Preprocessing, Translation, Data Mining, Interpretation/Evaluation

Data Mining: "The generation of useful knowledge from data".

Data

In this course Data is in:

- Standard (normalized) relations
- Relations with special data-types or extended expressivity
  - Spatial Types
  - Row Types
  - Collection Types
  - Inheritance
- Relations with special semantics over restricted expressivity
  - Temporal
  - Fact and Dimension Tables – Data Ware Housing
  - Description Logics - unary concepts/binary relations

Knowledge

Knowledge in this class:

- 'rules'
  - (Relation/Predicate/Concept) *classification* rules
  - Causal/Association rules
- Hierarchies clustering attribute domains.
- Coefficients and their corresponding 'basis' functions

Properties of Knowledge

Knowledge tends to be 'universally quantified formula' and thus has an infinite (or very large) extent in either its positive or negative form (or both).

Knowledge is *necessary* (laws of the domain) or *contingent* (accidental).

Knowledge is either *epistemic* or *ontological*.

Knowledge is derived either *inductively* or *deductively* (or is simply provided *a priori*).

Uses of Knowledge

Knowledge allows us to perform *diagnoses* or make *predictions* from information. Information = Data + Knowledge

Why would we want to do *diagnose* or *predict*?

- Prediction: What are the expected sales of item X?
- Diagnosis: What tends to occur before a surprise attack?

Data Mining Techniques

For a quick road-map to Data-Mining techniques consider various data forms paired with the possible target knowledge forms.

Note that the target Knowledge form places requirements on input Data.

Techniques often call for extra parameters to be supplied as well.

We shall consider:

- Mining of association rules.
- Generating classification rules over standard class labeled tuples.

Association Rules among Collections

Transaction	Basket
1	{milk, bread, juice}
2	{milk, juice}
3	{milk, eggs}
4	{bread, cookies, coffee}

An Association rule is: $X \Rightarrow Y$ where X at Y are sets of items and $X \cap Y = \emptyset$

For example $\{milk\} \Rightarrow \{juice\}$.

"67% of those who buy milk also buy juice"

67% is our confidence in this rule.

Support and Confidence

There are two important measures associated with a rule - the rule's *support* and the rule's *confidence*.

- support: the percentage of transactions where rule is *verified**.
- confidence: The proportion of transaction that verify the rule, to those that either falsify † or verify the rule.

$\{milk\} \Rightarrow \{juice\}$ has 50% support and 67% confidence.

$\{bread\} \Rightarrow \{juice\}$ has 25% support and 50% confidence.

Note that the $support(X)$ where X is a set of items is simply the proportion of the transactions that contain X over the whole sample.

*Both the antecedent and consequent sets are within the transaction

†The antecedent set, but not the consequent set is in the transaction

Probabilistic Characterization

$P(A|B) = \frac{P(A \cap B)}{P(B)}$

We may interpret the association rule $X \Rightarrow Y$ as:

$P(Y \subseteq S | X \subseteq S) = \frac{P(X \cup Y \subseteq S)}{P(X \subseteq S)}$

where S are the set of items in the transaction.

$P(X \cup Y \subseteq S)$ is the "support"

$P(Y \subseteq S | X \subseteq S)$ is the "confidence'

$P(X \subseteq S)$ is the 'coverage'

More shall be said about the probabilistic interpretation when we consider how to use association rules.

Fundamental Properties of Support

Property 1 *if* $support(X) > \beta$ *then* $support(Y) > \beta$ *where* $Y \subset X$.

Property 2 *if* $support(X) < \beta$ *then* $support(X \cup Y) < \beta$.

Obtaining 'Good' Association Rules

Over a set of m distinct objects there are $O(2^{2m})$ syntactically correct associations. (This bound is loose because X and Y are disjoint.)

We wish to only report interesting rules - those over a confidence threshold α and over a support threshold β.

Semi-Naive Algorithm

- 1.) Compute all sets* with sufficient ($> \beta$) support.
- 2.) Among each large item set X, consider the rules $X - Y \Rightarrow Y$ where $Y \subset X$:
  - If $\frac{support(X)}{support(Y)} > \alpha^{\dagger}$ then add $X - Y \Rightarrow Y$ to the answer set.

Still we have all $O(2^m)$ large item sets to consider.

*Termed the "large item sets"

†By property 1 such a rule will have proper support

The "Apriori" Algorithm

By exploiting property 2 we can improve this algorithm:

- 1.) Test support for item sets of size 1 - discard those with support less than β.
- 2.) Generate all distinct pairwise combinations of 1-item sets (generating the 2-item sets) - discard those with support less than β.
- 3.) Generate the $k-th$-item set by self joining the k-1-item sets $k-1$ - discard sets with insufficient support.
- 4.) Repeat until no additional items sets have sufficient support.

The "Apriori" Algorithm

Generates all of the large item sets.

Requires k scans of the database.

In addition we may store pre-computed large item-set supports for quick confidence calculations.

Using Association Rules

We must be cautious.

Assume that we generate two rules of sufficient support and confidence $\{beer\} \Rightarrow \{cigaretes\}$ and $\{beer\} \Rightarrow \{snus\}$.

Assume that rule $\{beer, snus\} \Rightarrow \{cigarettes\}$ has no support.

In an operational environment we may not blindly apply either rule to set X which contains $beer$.

Also what if we have the rule $\{snus\} \Rightarrow \{toothpaste\}$?

Then does $\{beer\} \Rightarrow \{toothpaste\}$? Not necessarily.

Practitioners should be aware of such limitations!

Association Rules among Hierarchies

We may obtain more general association rules if we group items into categories.

But we should obviously exclude the intra-hierarchy associations. $\{crest\} \Rightarrow \{toothpaste\}$

The inter-hierarchy associations may be interesting however $\{Duff-Beer\} \Rightarrow \{Pain-Killers\}$.

Negative Associations

"85% of people who buy yogurt do not buy soda".

Very hard in general to not get swamped with meaningless rules.

"100% of people who buy toothpicks do not buy Trocadero."

Using prior knowledge in the form of hierarchies can help determine what is an interesting rule.

"40% of people who by cigarettes buy beer, but 10% of people who buy Camels, do not buy Pripps".

You need access to the distribution of market share among beer and cigarettes to determine if this is unexpected, and hence interesting - what about using the sample itself to predict market share. Would this work?

Applying Association Rule Mining over Additional Data forms

You may apply these ideas to standard relations.

The problem may be considered over time dimensions and associations may be over subsequences – very difficult computationally.

Must calculate subsequences - must also track customers in the retail setting.

Discovery of Classification Rules

We are give a relational table with attributes. Of these attributes there exist a *dependent* attribute that we would like to classify tuples to based on a set of *predictor* attributes.

In a system with inheritance the class name can serve as the dependent attribute.

Given a sample of data in:

```
InsuranceRisk(ID) :- InsuranceInfo(ID, AGE, CARTYPE, GENDER)
 AGE > 16, AGE < 25, GENDER = 'Male', CARTYPE = 'Sports'.
```

Decision Trees

Each attribute on an internal node is the splitting attribute.

Each outgoing edge on an internal node has a predicate.

Each leaf is labeled with a dependent attribute.

Each path from root to leaf determines a classification rule.

Building a Decision Trees - Growth Phase

Top down greedy algorithm - growth phase

local computation of best splitting criteria (the attributes and predicates).

Database is partitioned along splitting criteria, algorithm recurs on subtrees.

Conclusions

We only scrated the surface of Data mining.

Other techniques to consider:

- Version Spaces