Integrated Framework of Enterprise Knowledge Management Based on Mining Association Rules

He Ying

School of mechanical engineering and automation, Beihang University,Beijing, China

AECC Hunan Aviation Power Plant Research Institute, Zhuzhou, China

Key words: integrated framework; knowledge management; mining association rules; enterprise knowledge management system

Abstract: An Enterprise requires fund of knowledge to choose the right processes, technologies and resources. Enterprise knowledge management gradually become the key to victory in the fierce competition. Associations rules is top ten algorithm and an important data mining model studied extensively by the database and data mining community. Association rules are applied to the enterprise management in this paper and acquire expected result .Finally, enterprise knowledge management system was proposed which was based on the integrated framework.

1 Introduction

Enterprise knowledge referring to the ability of an organization to precisely define, easily integrate and effectively retrieve data for both internal applications and external communication. Enterprise knowledge focused on the knowledge of accurate, consistent and transparent content. Enterprise knowledge emphasizes knowledge effectiveness, precision, granularity and meaning and is concerned with how the content is integrated into enterprise applications as well as how it is passed along from one enterprise process to another.

In data mining, association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness. Based on the concept of strong rules, association rules were proposed by Agrawal et al in 1993. Today association rule application in many areas including business, web usage mining, intrusion detection, Continuous production and bioinformatics.

This paper build a integrated framework and use WEKA open source library directly to implement the association rule mining algorithm that search enterprise knowledge in the enterprise knowledge database. Integrated framework have a great practical value that can application in various software development.

2 Theory introduction

2.1 Mining Association Rules

There are a large number mining association rules algorithms, they use different strategies and data structure and their resulting sets of rules are all the same.

Resulting sets of rules model is described as follows(Agrawal, 2013):

$I = \{i_1, i_2, \ldots, i_m\}$ is a set of items.Transaction t is a set of items, and $t \subseteq I$.Transaction Database T is a set of transactions $T = \{t_1, t_2, \ldots, t_n\}$.A transaction t contains X, a set of items (itemset) in I, if $X \subseteq t$.

An association rule is an implication of the form:$X \rightarrow Y$, where X, $Y \subset I$, and $X \cap Y = \varnothing$ .An itemset is a set of items and a k-itemset is an itemset with k items.

The rule holds with support s in T (the transaction data set) if s% of transactions contain $X \cup Y$. $s = Pr(X \cup Y)$. The rule holds in T with confidence c if c% of transactions that contain X also

contain Y.conf = Pr(Y | X).An association rule is a pattern that states when X occurs, Y occurs with certain probability.

The support count of an itemset X, denoted by X.count, in a data set T is the number of transactions in T that contain X. Assume T has n transactions. Then, for the rule $X \rightarrow Y$,

$$support = \frac{(X \cup Y).count}{n} \quad (1)$$

$$confidence = \frac{(X \cup Y).count}{X.count} \quad (2)$$

The goal of algorithm is find all rules that satisfy the user-specified minimum support (minsup) and minimum confidence (minconf).

The support of an itemset X in a transaction Database T containing n transactions is defined as:

$$support = \frac{X.count}{n} \quad (3)$$

A frequent itemset is an itemset whose support is ≥ minsup.

2.2 The Apriori Algorithm

There are a large number of mining association rules. They use different strategies and data structures, but their resulting sets of rules are all the same that given a transaction data set T, and a minimum support and a minimum confident, the set of association rules existing in T is uniquely determined.

Any algorithm should find the same set of rules although their computational efficiencies and memory requirements may be different. This paper study the apriori algorithm for it is the best known algorithm mining association rules.

There are two steps(Agrawal, 2013; Mannila, 2012; Savasere, 2000; Park, 2003; Toivonen, 1996; Brin, 2001; Sarawagi, 1999): first,find all itemsets that have minimum support (frequent itemsets, also called large itemsets). Second, use frequent itemsets to generate rules.

Mining all frequent itemsets' key idea is the apriori property (downward closure property) what any subsets of a frequent itemset are also frequent itemsets. Details are follows:

Step1:Iterative algo (also called level-wise search) what find all 1-item frequent itemsets; then all 2-item frequent itemsets, and so on. Its steps are:

- generate length (k+1) candidate itemsets from length k frequent itemsets, and
- test the candidates against DB

Step2:Find frequent itemsets of size 1: F_1

Step3:From k = 2,C_k = candidates of size k: those itemsets of size k that could be frequent, given F_{k-1}; F_k = those itemsets that are actually frequent, $F_k \subseteq C_k$ (need to scan the database once).

The candidate-gen function takes F_{k-1} and returns a superset (called the candidates) of the set of all frequent k-itemsets. It has two steps:

- join step: Generate all possible candidate itemsets C_k of length k
- prune step: Remove those candidates in C_k that cannot be frequent.

Generating rules from frequent itemsets(Han, 2006 ; Agarwal, 2002 ; Grahne, 2003 ; Liu, 2006) is Frequent itemsets ≠ association rules and one more step is needed to generate association rules.For each frequent itemset X, for each proper nonempty subset A of X, let B = X - A ,$A \rightarrow B$ is an association rule if confidence($A \rightarrow B$) ≥ minconf, support($A \rightarrow B$) = support($A \cup B$) = support(X) and confidence($A \rightarrow B$) = support($A \cup B$) / support(A).

If a subset X of a frequent itemset L does not generate a rule, then no need to consider generating rules from subset of X.If a rule (L-c) → c holds, then all the rules of the form (L-x) → x also holds, where x is a non-empty subset of c.

In summary, Generating rules is to recap, in order to obtain $A \rightarrow B$, we need to have support($A \cup B$) and support(A);all the required information for confidence computation has

already been recorded in itemset generation. No need to see the data T anymore; this step is not as time-consuming as frequent itemsets generation.

3 Integrated framework

Integrated framework mainly adopts template, component technology in the web development, so as to ensure the maximum flexibility and scalability. In the process of application, achieve optimization component design in many aspects such as cache, flow control, file processing, graphics, etc. so it can be said, integrated framework has its own unique advantages.

Compared with the existing three layer open source framework, integrated framework's main features include:

Unified transaction management that ensure a transaction to form a single operation, if the operation is abnormal, it will roll back the transaction, only all operations success it will commit the transaction, in order to ensure the accuracy of the business logic.

Unified connection management, page level connection based on a unified management that ensure the page at the start of the operation for the connection, the connection is automatically released logout page, to avoid leakage resulting collapse pool.

Unified exception management that ensure all the error messages generated, it can be fully reflected in the final pages and ensure timely tracking and correct positioning error, ensure the stability of the program.

Component based design that can easily maintain the old version and the new version.

Based on the configuration parameters, for different enterprise knowledge module, member enterprises of different subsystems, provide flexible configuration parameters, reflect the needs of the application of different.

Flow configuration based on different members, according to enterprise users, depending on the configuration of the enterprise process, provide demand difference based on unified version

Provides log tracking components, can be in accordance with the flexible search the specified log, convenient maintenance location problem.

Component and reusable technology is the key of integrated framework. There are seven layers in integrated framework, every layer has own unique function. The reusable component technology is shown as fig.1:

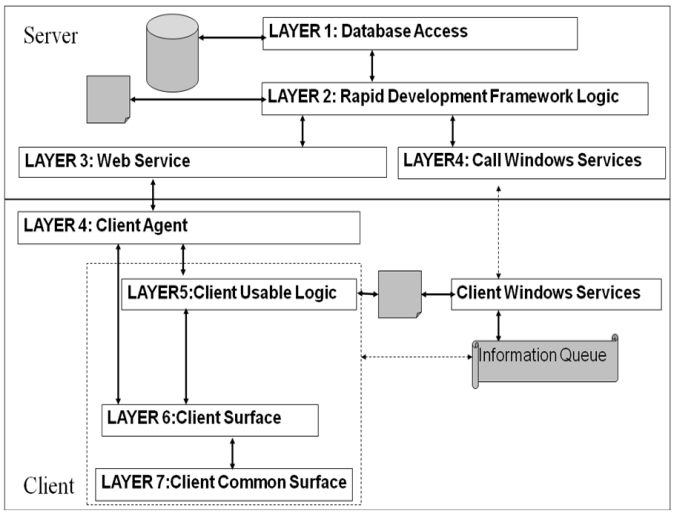

Figure 1. Multi-layer framework structure

4 Implementation of enterprise knowledge management system based on integrated framework

First, implement the association rule mining on Weka employing the Association Rule discovery algorithm. Specifically, a data set named enterpriseknowledge.txt.
Weka's data format is special and need prepare in advance. On weka, the standard format of data set can be recognized is .csv or .arff. Therefore, in the first, enterpriseknowledge.txt should be transformed to .arff.

Second, implement integrated framework's configuration, there are four steps.

1) Integrated framework run environment configuration

a) Install and configure eclipse,JDK,ant,weblogic and Integrated framework, run build in eclipse of ant to initialization.

b) Run weblogic http://localhost:7001/console .

c) Install and deploy web application in weblogic.

2) Integrated framework run flow

a) Web application search deploy describe symbol(web.xml) as entrance to run program, and loading servlet,filter,monitor and session.

b) Client post visit path and send request to server, server search client vist servlet in web.xml and then find app servlet.

c) Find configuration file .application that has the same name with app servlet .

d) Loading .application's configuration such as engine configuration,visit configuration,page configuration,etc.

e) According to client post page's name to search page file in .application.。

f) Through page file search java,html file.

3) WEB application deploy configuration

```
<filter>
<filter-name>redirect</filter-name>
<filter-class>org.apache.tapestry.RedirectFilter  </filter-class>
</filter>
<filter-mapping>
<filter-name>redirect</filter-name>
<url-pattern>/</url-pattern>
</filter-mapping>
<servlet>
<servlet-name>quickstart</servlet-name>
<servlet-class> org.apache.tapestry.ApplicationServlet</servlet-class>
<load-on-startup>0</load-on-startup>
</servlet>
<servlet-mapping>
<servlet-name>quickstart</servlet-name>
<url-pattern>/app</url-pattern>
</servlet-mapping>
<session-config>
<session-timeout>1800</session-timeout>
</session-config>
```

4) System whole situation configuration

```
<project
    name="EnterpriseKnowledge"
    productmode="false"
    uploadpath="upload"
    version="1.5.1"
```

```
        model=“HY”
        logintype=“WEB” />
    <tuxconn type="WTC"/>
    <subsys default="sale">
        <DynamicKnowledge addr="http://191.118.0.1:7000/"/>
        < StaticKnowledge addr=" http://191.118.0.2:7000/"/>
    </subsys>
    <database default="ehy">
        <sdcrm1 type="EK“ EK =“RDF"/>
        <sdcen1 type=" EK “ EK =" RDF "/>
    </database>
```

Third, write source code of page template (.html), page specification (.page),page class(.jave) and some component class related to the business logic in enterprise knowledge management system. The following are several instances illustrate the writing method:

```
    <- system page template example ->
      <html xmlns="http://www.w3.org/1999/xhtml">
      <head SCKid="@ RDF:Head">
      <meta http-equiv="Content-Type" content="text/html; charset=gbk"/>
      <title> EnterpriseKnowledge </title>
      <link href="/component/styles/styles_all.css" rel="stylesheet" type="text/css"
media="screen"/>
      </head>
      <body EKid ="@Body">
      <form EKid ="@Form">
        ...
      </form>
      </body>
      </html>

    <- system page  specification example->
    <page-specification class="com.linkage.quickstart. EnterpriseKnowledge ">
    <property-specification name="condition" type="com. framework.IData"/>
    </page-specification>

    <- system page  class example ->
      public abstract class EnterpriseKnowledge extends AppPage {
      /* define enterprise knowledge object in page template */
      public abstract void setInfos(IDataset infos);
      public void queryEnterpriseKnowledge (IReq cycle) throws Exception {
      /* enterprise knowledge query */
      PageData pd = getPageData();
      IDataset infos = ...(enterprise knowledge);
      setInfos(infos);
        }
      }
```

5 Conclusion

The application of integrated framework can solve some some enterprise knowledge management problems including sharing of information flow and workflow and high efficiency operation. But

integrated framework lack of innovation and decision analysis models, so enterprise knowledge management system have not innovation and decision analysis, this is future effort direction.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. SIGMOD'93.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. VLDB'94.

[3] H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. KDD'94.

[4] Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. VLDB'95.

[5] J. S. Park, M. S. Chen, and P. S. Yu. An effective hash-based algorithm for mining association rules. SIGMOD'95.

[6] H. Toivonen. Sampling large databases for association rules. VLDB'96.

[7] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket analysis. SIGMOD'97.

[8] S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with relational database systems: Alternatives and implications. SIGMOD'98.

[9] J. Han, J. Wang, Y. Lu, and P. Tzvetkov. Mining Top-K Frequent Closed Patterns without Minimum Support. ICDM'02.

[10] R. Agarwal, C. Aggarwal, and V. V. V. Prasad. A tree projection algorithm for generation of frequent itemsets. J. Parallel and Distributed Computing:02.

[11] G. Grahne and J. Zhu, Efficiently Using Prefix-Trees in Mining Frequent Itemsets, Proc. ICDM'03 Int. Workshop on Frequent Itemset Mining Implementations (FIMI'03), Melbourne, FL, Nov. 2003

[12] H. Liu, J. Han, D. Xin, and Z. Shao, Mining Interesting Patterns from Very High Dimensional Data: A Top-Down Row Enumeration approach, SDM'06.

[13] Luigi T.De Luca, Propulsion physics (EDP Sciences, Les Ulis, 2009)

[14] F. De Lillo, F. Cecconi, G. Lacorata, A. Vulpiani, EPL, 84 (2008)