Integrating Inductive and Deductive Reasoning for Database Mining

Evangelos Simoudis Brian Livezey Randy Kerber
Lockheed Artificial Intelligence Center
O/96-20 B/254F
3251 Hanover Street
Palo Alto, CA 94304
{simoudis, livezey, kerber}@aic.lockheed.com

Abstract

Database mining is the process of finding previously unknown rules and relations in large databases. Often, several database mining techniques must be used cooperatively in a single application. In this paper we present the *Recon* database mining framework, which integrates three database mining techniques: rule induction, rule deduction, and data visualization.

1 Introduction

Database mining is the process of finding previously unknown rules and relations in large databases. The extracted information can be used to form a prediction or classification model, identify trends and associations, refine an existing model, or provide a summary of the database(s) being mined. A number of database mining techniques, e.g., rule induction, neural networks, and conceptual clustering, have been developed and used individually in domains ranging from space exploration [2] to financial analysis [10]. Frequently, several techniques must be employed cooperatively to support a single database mining application. In this paper we discuss how rule induction, deductive database processing, and data visualization techniques can be used cooperatively to create rule-based models by mining the contents of relational databases. We also present the *Recon* database mining framework which integrates these three techniques.

The operations performed during database mining are discussed in Section 2. *Recon*'s architecture is described in Section 3. Issues relating to the distribution of data among *Recon*'s database mining modules are discussed in Section 4. The interaction among the three database mining techniques used in *Recon* are described in Section 5. A model-creation example from the domain of stock portfolio creation is presented in Section 6. Related work is discussed in Section 7, and conclusions are presented in Section 8.

2 Database Mining and Model Creation

Models created by mining a database can be statistical, neural, or symbolic, depending on the mining technique used. The configuration of *Recon* described in this paper is tailored to produce symbolic models.

A symbolic model consists of a set of "if ... then ..." rules. The quality of each rule is measured by its predictive ability, its statistical validity, its generality, and its interestingness. For example, consider the rule "if the revenues of a publicly-traded corporation are five times its profits for three consecutive quarters, then the return on investment of its stock during the next quarter will be greater than 20%." This rule will be classified as (1) very predictive if, for example, it correctly predicts the return on investment for eight out of the ten stocks, (2) statistically valid, if a statistical significance test indicates that its predictiveness could not easily have occurred by chance, (3) very general if it is very predictive during several time periods and is relevant for a large number of stocks, and (4) very interesting if it provides a useful insight to the analyst. The rule "the return on investment of stocks issued by pharmaceutical companies is greater than 20%" may have been very predictive during the summer of 1992 but not at present. The rule is not very general because it only applies during a single time period and it only characterizes stocks of pharmaceutical companies, a small percentage of the stocks traded in the various exchanges. It may, however, be very interesting if the stocks of pharmaceutical companies do not usually provide high returns on investment.

The mining of such rules is a process of incremental refinement performed by interweaving three operations: (1) testing and refinement of rules hypothesized by the analyst, (2) automatic discovery of rules by the database mining system, and (3) integration of the best of the hypothesized and discovered rules into a cohesive model.

Analysts often hypothesize value-prediction and classification rules as well as trends and associations. For example, an investment analyst may have hypothesized the first rule above. Since the quality of this rule is not known *a priori*, the rule must be tested against historical data. The test results allow the user to establish the rule's quality. The testing could lead the analyst to accept the rule and incorporate it into the model being developed, refine the rule, or reject it altogether. In some cases, such analysis may lead the analyst to discover erroneous data.

Analysts cannot hypothesize all the rules that might be included in a model. The data mining system explores the contents of the database to automatically identify rules supported by the data. For example, by exploring the target database, a rule induction component may form a rule stating that "if a company's earnings per share growth is greater than 50% during six consecutive quarters, then the return on investment of its stock during the quarter following this period will be greater than 20%." The rules resulting from automatic discovery can be ranked according to their discriminating power. The discovered rules can also be tested on a different database so that their predictive ability and generality can be established. Using all of the provided information, the user selects a subset of the discovered rules, possibly edits them, and incorporates them into the model being developed.

3 Recon's Architecture

Recon consists of: a command module, a server, one or more database mining modules, a knowledge repository, and interfaces to relational database management systems. In *Recon* rule validation is performed using the IDEA deductive database processor [4]. Rule discovery is

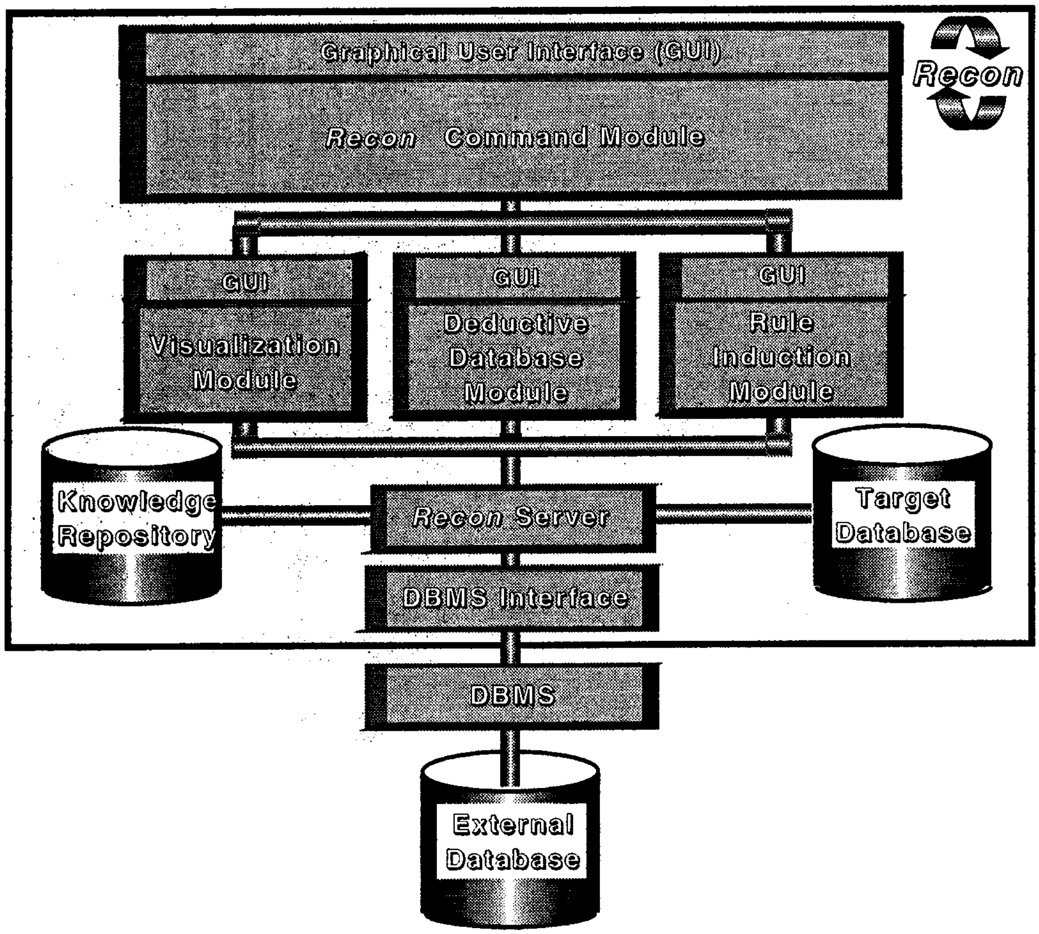


Figure 1: The architecture of the *Recon* database mining framework

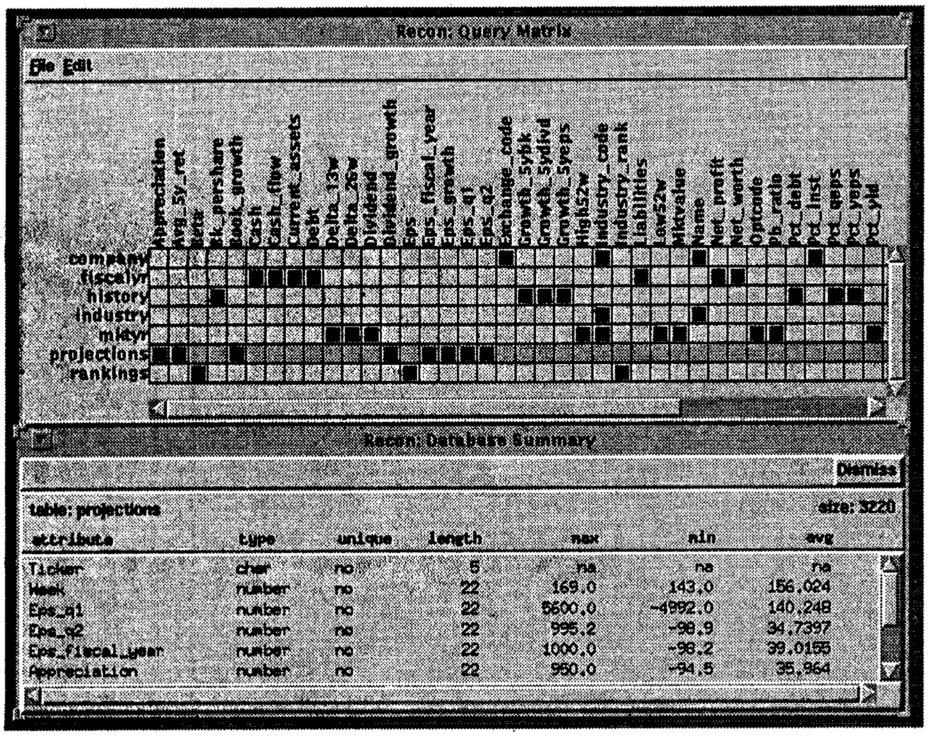
performed using the OTIS rule induction system [5] and a data visualization system [8]. *Recon*'s architecture is shown in Figure 1.

The analyst uses the command module to connect to a particular database. The server interacts with the database management system of the connected database by issuing Structured Query Language (SQL) commands through the database Application Programming Interface (API). The *Recon* server distributes data and knowledge among the connected database and the database mining modules.

4 The Recon Server

Recon's data and knowledge distribution operations are performed by the *Recon* server. The server maintains the target database, allows the user to direct data from the deductive database module to the other database mining modules, maintains the knowledge repository, and distributes stored knowledge and data to the *Recon* modules.

The target database is created and refined through interaction between the user and the *Recon* server. In particular, after the user connects to a database, the *Recon* server automat-



	Appreciation	Avg_5y_ret	Beta	Bk_pershare	Book_growth	Cash	Cash_flow	Current_assets	Debt	Delta_13w	Delta_26w	Dividend	Dividend_growth	Eps	Eps_fiscal_year	Eps_growth	Eps_q1	Eps_q2	Exchange_code
company	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	■
fiscalyr	□	□	□	□	□	■	■	■	■	□	□	□	□	□	□	□	□	□	□
history	□	□	□	■	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
industry	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
mktyr	□	□	□	□	□	□	□	□	□	■	■	■	□	□	□	□	□	□	□
projections	■	■	□	□	■	□	□	□	□	□	□	□	■	□	■	■	■	■	□
rankings	□	□	■	□	□	□	□	□	□	□	□	□	□	■	□	□	□	□	□

	Growth_5ybk	Growth_5ydivd	Growth_5yeps	High52w	Industry_code	Industry_rank	Liabilities	Low52w	Mktvalue	Name	Net_profit	Net_worth	Optcode	Pb_ratio	Pct_debt	Pct_inst	Pct_qeps	Pct_yeps	Pct_yld
company	□	□	□	□	■	□	□	□	□	■	□	□	□	□	□	■	□	□	□
fiscalyr	□	□	□	□	□	□	■	□	□	□	■	■	□	□	□	□	□	□	□
history	■	■	■	□	□	□	□	□	□	□	□	□	□	□	■	□	■	■	□
industry	□	□	□	□	■	□	□	□	□	■	□	□	□	□	□	□	□	□	□
mktyr	□	□	□	■	■	□	□	■	■	□	□	□	■	■	□	□	□	□	■
projections	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
rankings	□	□	□	□	□	■	□	□	□	□	□	□	□	□	□	□	□	□	□

table: projections size: 3220

attribute	type	unique	length	max	min	avg
Ticker	char	no	5	na	na	na
Week	number	no	22	169.0	143.0	156.024
Eps_q1	number	no	22	5600.0	-4992.0	140.248
Eps_q2	number	no	22	995.2	-98.9	34.7397
Eps_fiscal_year	number	no	22	1000.0	-98.2	39.0155
Appreciation	number	no	22	950.0	-94.5	35.964

Figure 2: Schema information of a database with corporate stocks

ically extracts the database's schema and displays it in the Query Matrix. A portion of the extracted information from a database of corporate stocks is shown in Figure 2.

The Query Matrix presents the database's tables (base relations) along the vertical axis and attributes along the horizontal axis. The squares in the matrix indicate which attributes appear in which tables. More detailed information about a particular table can be seen by selecting a relation in the Query Matrix.

Using the base relations from the Query Matrix, the analyst can employ the deductive database module to define new domain-specific concepts; this process is described in Section 5.2. *Recon*'s server automatically incorporates each concept into the target database's schema, displaying it in the Query Matrix, so that it can be used seamlessly with the base relations in subsequent queries or concept definitions. These definitions can be sent to the rule induction module to be incorporated as antecedents in the discovered rules.

Often, analysts use *Recon*'s database mining modules on only a subset of the database. These subsets, called **data sources**, are defined by querying the target database. Data sources are maintained by the *Recon* Server and can be stored explicitly by saving the results of the query, or implicitly by saving the query itself. The *Recon* server can randomly sample a data source if further reduction is necessary. In Figure 3, the user has selected the data source called "solvent" and indicated that the data source should be loaded into the visualization module.

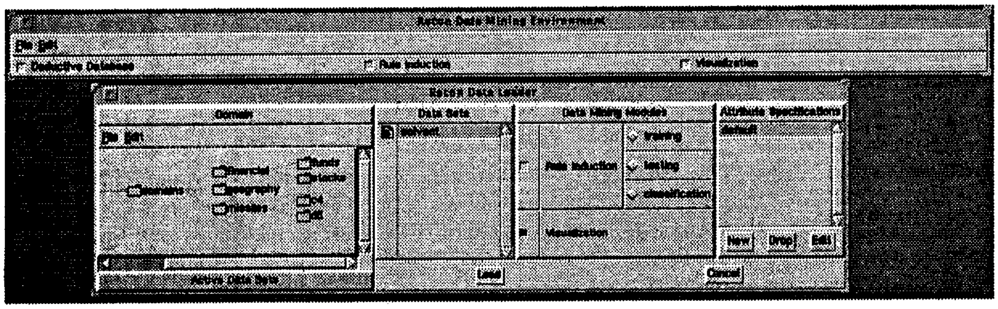


Figure 3: Selecting a data source using the *Recon* server

5 Deduction, Induction, and Visualization

In this section we describe how *Recon*'s three database mining modules can be used cooperatively to create a rule-based classification model. In a typical scenario, the analyst first gets a "feel" for the contents of the target database using the visualization module. In the process, he may identify relations between various attributes which he proceeds to encode using the deductive database module. The analyst invokes the rule induction module, provides it with a sample of the target database over which to discover rules, and initiates the discovery process. While the rule discovery operation proceeds in the background, the analyst hypothesizes and tests other rules using the deductive database module. As part of the testing process, data generated by the deductive database module may be sent to the visualization module for more detailed exploration. At any point, the analyst may (1) interrupt the rule discovery operation, (2) use the *Recon* server to import into the rule induction module encoded concepts, and/or validated rules from the knowledge repository, thereby "seeding" this module's knowledge base, and (3) restart the induction process. The user selects a subset of the induced rules to be stored in the knowledge repository.

5.1 Data visualization

Data visualization is particularly appropriate both for obtaining a global view for a data set and also for noticing important phenomena that hold for a relatively small subset of a data set. Such phenomena often tend to be "drowned out" by the rest of the data when statistical methods are used. Furthermore, the analyst does not need to know specifically the type of phenomena he is looking for in order to notice something interesting. In addition, when used on data with distributions that are not well-behaved, visualizations tend to be more effective than many statistical methods, because the latter often make limiting assumptions about the data distributions.

The visualization process begins with the user selecting a data source from the *Recon* server and importing it into the visualization module. *Recon*'s visualization module permits the analyst to:

- Mark subsets of the data set that are of particular interest. Each such subset is plotted using a different color.
- Focus on a portion of the displayed data. For example, Figure 4 shows a simple visualization that includes two scatter plots produced by *Recon*'s visualization module created

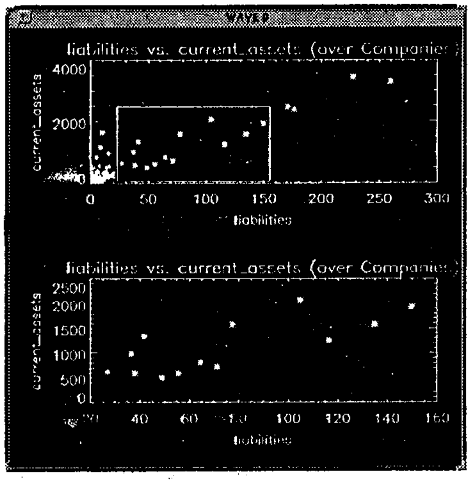


Figure 4: Visualization of data about publicly traded companies

in order to determine the relationship between current-assets and liabilities of solvent companies. The top scatter plot includes the entire contents of the data source (see Figure 3). The bottom scatter plot provides more detail of the area indicated by the rectangle on the top scatter plot. In this case, it was selected to include companies that show an almost linear relationship between current-assets and liabilities.

- Select a portion of the data displayed in one visualization, e.g., a two-dimensional scatter plot, and visualize the values of attributes of interest using another visualization, e.g., a two-dimensional line plot. For example, after establishing, through the use of a two-dimensional scatter plot, that there exists an almost-linear relationship between the values of the current assets and current liabilities attributes, the analyst can select the data points that obey this relationship and display the values of their quarterly earnings over a period of a year using a two-dimensional line plot.
- Select a subset of the data to export to another *Recon* module for further analysis.

5.2 Deductive database processing

Recon's deductive database module is used for formulating queries, defining **concepts**, or derived relations, representing rules to be validated, and refining existing concepts and rules.

A concept is defined in terms of its name, a set of attributes, and one or more relations among the attributes. An attribute may either belong to one of the database's base relations, or be part of another, previously defined, concept. The user selects the appropriate relations and/or concepts from the Query Matrix and relates the attributes by specifying equalities and inequalities among the attributes. The new concept is incorporated into the Query Matrix and can be used in future queries, concept definitions, and rules.

A database that contains such user-defined knowledge is called a **deductive database**. The values of concepts are not stored in the database. They are computed dynamically via deductive inferences in response to a user's query. In particular, after a query is formulated by the user it is automatically expanded by the deductive database module until it consists solely of base relations and computable functions. The deductive database module transforms each expanded query into a set of optimized SQL expressions. The expanded query is then posed to the target database.

The deductive database module provides justifications for query responses. The justifications detail the exact chain of reasoning that led to the result in question. In this way, the user can better understand the results and determine whether the definitions of particular concepts need to be modified/refined, or whether the data in the target database is anomalous.

The analyst also uses the deductive database module to form "if... then ..." rules. The antecedents and consequents of these rules may contain a mixture of user-defined concepts and base relations. The deductive database module expands the included concepts, as was described above, and automatically creates two queries which, after optimization, are posed to the target database. The first query returns the set of records that support the rule. The second query returns the set of records that match the rule's left-hand side but do not match its right-hand side. From the returned information the analyst determines the rule's discrimination ability, thus starting to establish the rule's quality. The user can also obtain a justification of why each answer was included in each of the two returned sets. To this end, the deductive database module displays, in the form of a proof tree, the concept-reduction process it followed. The user can view the definition of each concept included in this justification by selecting the desired concept in the proof tree.

5.3 Rule induction

While the deductive database module allows the analyst to express knowledge and test it against the target database, rule induction is used to automatically explore the target database to discover rules that characterize its contents. The data presented to the rule induction module must belong to a small set of classes that have been predefined by the user. For example, assume that the target database contains historical data about the quarterly performance of stocks, including the quarterly return on investment of each stock. Further assume that the analyst defines two classes: "high return on investment," for stocks with quarterly return on investment greater than 20%, and "low return on investment," for the rest of the stocks. The induced rules express generalizations over the input data that are useful for distinguishing among these classes.

Recon's rule induction module is based on the OTIS rule induction system [5]. It was selected for inclusion into *Recon* over other such algorithms for three reasons. First, it is able to produce high quality rules even when the input data is noisy and incomplete. Second, it generates "if ... then ..." rules, the same representation used by *Recon*'s deductive database module. Finally, it is able to accept knowledge defined using the deductive database module. Such domain knowledge expedites the rule discovery task while simultaneously improving the quality of the induced rules.

Since it is rarely possible to find a single, perfect discriminating rule that matches all examples in one class and no examples in any other class, *Recon*'s rule induction module produces a collection of rules, each of which is assigned a strength factor to indicate how well it predicts the target attribute. This factor is one of the measures used by the analyst to establish each rule's quality.

The user can establish the predictive ability and generality of the induced rules by testing them against the contents of another data source imported from the *Recon* server. The user can request an explanation of why a particular prediction was made. In response to such a request, the rule induction module displays the subset of the rules that contributed to that prediction. After examining each rule and the weight it provided to the prediction, the analyst may decide to ignore certain attributes from the input data set or refine the definitions of concepts using the deductive database module. After making modifications, the analyst can re-test the rule set and store some or all of the rules in the knowledge repository.

6 Using Recon for Stock Portfolio Creation

We now provide a simple example of how an investment analyst can use *Recon* to create a rule-based model for selecting stocks that yield a high return on investment. Creating rule-based stock selection models is a very appropriate domain for database mining because of the large number of stocks from which the money manager can select to create a portfolio and the vast amount of historical data about each stock. The use of database mining techniques allows the analyst to consider many more factors and much more data, thus leading to the development of more accurate value-prediction models.

We assume that the investment analyst has access to a database with monthly data about a universe of stocks. The process begins by creating the target database through *Recon*'s server. Assume that the target database contains data about 1500 stocks over a period of seven years. The user's goal is to create a portfolio of approximately 100 stocks.

The user sends the target database, in its entirety, to the visualization module. In order to investigate whether there exists a relationship between the values of the "Current PE" (current price-earnings ratio) and "Recent price" attributes, the user creates a two-dimensional scatter plot and establishes that there exists a linear relationship between the values of these two attributes.

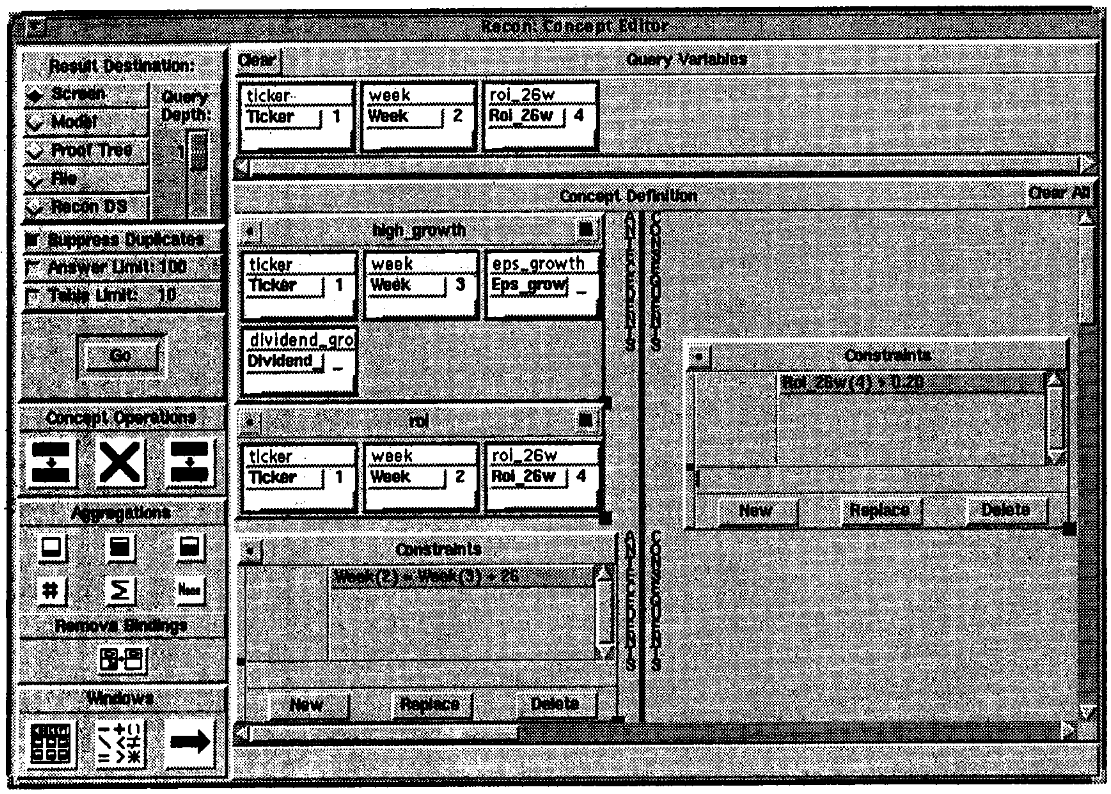
Next, the analyst uses the deductive database module to interactively test *his* knowledge about indicators that can discriminate stocks which have the potential of providing high return on investment. The user hypothesizes that earnings-per-share growth and dividend growth are good indicators. He defines the concept *high-growth* to characterize stocks whose *earnings-per-share-growth* and *dividend-growth* are both greater than 50%.

The analyst uses the deductive database module to formulate rules and test them against the target database. An example rule states that "**If** a stock is *high-growth* at time t, **then** its *return on investment* two quarters later will be greater than 20%." The definition of this rule is shown in Figure 5.

Once the user instructs *Recon* to test the stated rule, the system retrieves from the database two sets of answers: the records (stocks) that support the rule and those which refute it. After inspecting the returned information, the user decides that he has discovered a useful indicator for high return on investment, i.e., the high-growth concept, and a useful rule. He incorporates both of them into the model being developed. The rule is sent to the *Recon* server where it is entered into the knowledge repository.

While the analyst interactively creates indicators and tests rules, *Recon*'s rule induction module automatically discovers rules by exploring a randomly selected sample of the target database. The analyst interrupts the rule induction process and requests the rules discovered thus far. Before presenting them to the user, the rule induction module sorts the discovered rules based upon their discrimination power, determined by the percentage of the instances from each class in the target database the rule matches. Figure 6 shows a subset of the induced rules. The columns labeled "High" and "Low" indicate what percentage of each of the two sets the rule matched.

The user tests the discovered rules by applying them to another randomly selected data set. For each stock in this set, the rule induction module displays its return on investment



■ Suppress Duplicates
Answer Limit: 100
Table Limit: 10

ticker	week	roi_26w
Ticker 1	Week 2	Roi_26w 4

high_growth

ticker	week	eps_growth
Ticker 1	Week 3	Eps_grow _
dividend_gro		
Dividend _		

roi

ticker	week	roi_26w
Ticker 1	Week 2	Roi_26w 4

Constraints: Week(2) = Week(3) + 26

Constraints: Roi_26w(4) > 0.20

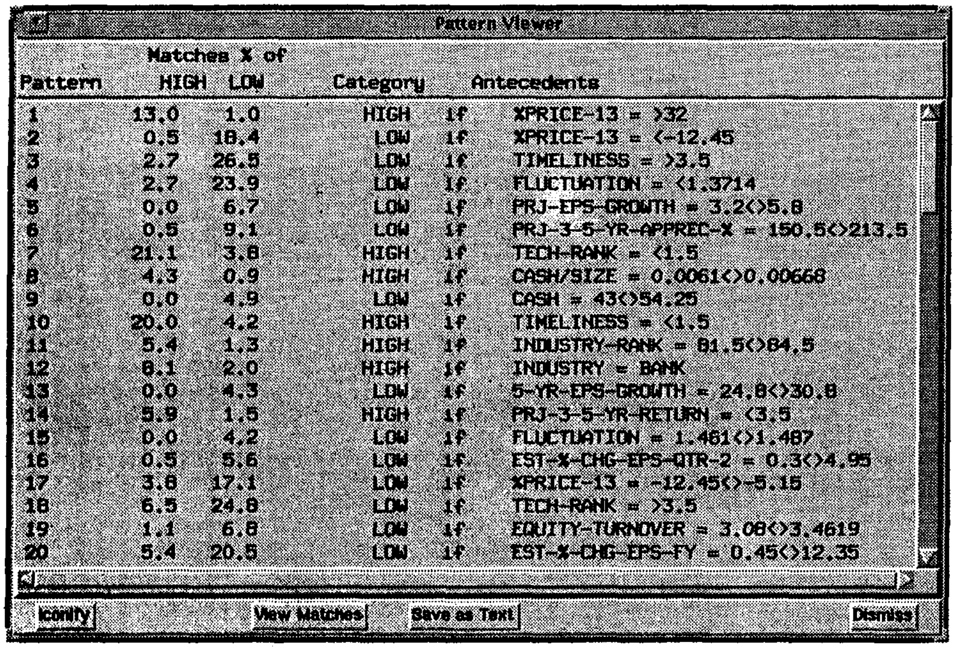
Figure 5: A rule expressed using *Recon*'s deductive database module

prediction, along with the confidence level associated with the prediction, as well as the stock's actual return on investment.

The user can select individual stocks from this list and request the rationale for the associated prediction. Of particular interest are those cases where the wrong prediction was made with a high degree of confidence. The explanation consists of the set of rules that were used to make the particular prediction. For example, the explanation for the stock with ticker symbol STL is shown in Figure 7. Associated with each rule is the weight that the rule contributed to the prediction. The sum of the individual rule weights determines the overall confidence in the prediction.

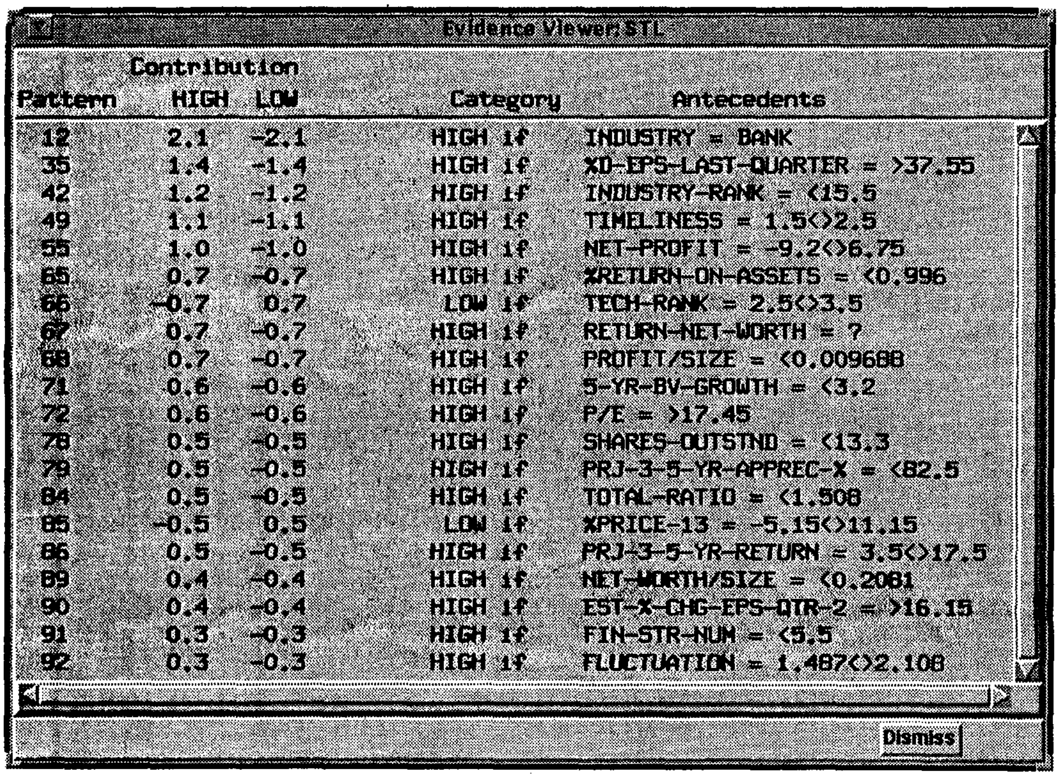
By examining the evidence for incorrect predictions, the user can identify low-quality rules. In this case, the user decides to delete the attribute INDUSTRY from the target database because it was only relevant during the time period covered in the target database and is not a generally useful indicator. The user restarts the rule induction process over the revised target database. Finally, the user selects the desired set of discovered rules and sends them to the *Recon* server where they are incorporated into the knowledge repository.

The model developed through the interweaving of these three operations: visualization, deductive database processing, and rule induction is then applied on a universe of current stocks. The top 100 of the stocks for which the model predicts a high return on investment are included in the portfolio.



	Matches X of				
Pattern	HIGH	LOW	Category		Antecedents
1	13.0	1.0	HIGH	if	XPRICE-13 = >32
2	0.5	18.4	LOW	if	XPRICE-13 = <-12.45
3	2.7	26.5	LOW	if	TIMELINESS = >3.5
4	2.7	23.9	LOW	if	FLUCTUATION = <1.3714
5	0.0	6.7	LOW	if	PRJ-EPS-GROWTH = 3.2<>5.8
6	0.5	9.1	LOW	if	PRJ-3-5-YR-APPREC-% = 150.5<>213.5
7	21.1	3.8	HIGH	if	TECH-RANK = <1.5
8	4.3	0.9	HIGH	if	CASH/SIZE = 0.0061<>0.00668
9	0.0	4.9	LOW	if	CASH = 43<>54.25
10	20.0	4.2	HIGH	if	TIMELINESS = <1.5
11	5.4	1.3	HIGH	if	INDUSTRY-RANK = 81.5<>84.5
12	8.1	2.0	HIGH	if	INDUSTRY = BANK
13	0.0	4.3	LOW	if	5-YR-EPS-GROWTH = 24.8<>30.8
14	5.9	1.5	HIGH	if	PRJ-3-5-YR-RETURN = <3.5
15	0.0	4.2	LOW	if	FLUCTUATION = 1.461<>1.487
16	0.5	5.6	LOW	if	EST-%-CHG-EPS-QTR-2 = 0.3<>4.95
17	3.8	17.1	LOW	if	%PRICE-13 = -12.45<>-5.15
18	6.5	24.8	LOW	if	TECH-RANK = >3.5
19	1.1	6.8	LOW	if	EQUITY-TURNOVER = 3.08<>3.4619
20	5.4	20.5	LOW	if	EST-%-CHG-EPS-FY = 0.45<>12.35

Figure 6: Rules discovered by *Recon*'s rule induction module



	Contribution			
Pattern	HIGH	LOW	Category	Antecedents
12	2.1	-2.1	HIGH if	INDUSTRY = BANK
35	1.4	-1.4	HIGH if	XD-EPS-LAST-QUARTER = >37.55
42	1.2	-1.2	HIGH if	INDUSTRY-RANK = <15.5
49	1.1	-1.1	HIGH if	TIMELINESS = 1.5<>2.5
55	1.0	-1.0	HIGH if	NET-PROFIT = -9.2<>6.75
65	0.7	-0.7	HIGH if	%RETURN-ON-ASSETS = <0.996
66	-0.7	0.7	LOW if	TECH-RANK = 2.5<>3.5
67	0.7	-0.7	HIGH if	RETURN-NET-WORTH = ?
68	0.7	-0.7	HIGH if	PROFIT/SIZE = <0.009688
71	0.6	-0.6	HIGH if	5-YR-BV-GROWTH = <3.2
72	0.6	-0.6	HIGH if	P/E = >17.45
78	0.5	-0.5	HIGH if	SHARES-OUTSTND = <13.3
79	0.5	-0.5	HIGH if	PRJ-3-5-YR-APPREC-X = <82.5
84	0.5	-0.5	HIGH if	TOTAL-RATIO = <1.508
85	-0.5	0.5	LOW if	%PRICE-13 = -5.15<>11.15
86	0.5	-0.5	HIGH if	PRJ-3-5-YR-RETURN = 3.5<>17.5
89	0.4	-0.4	HIGH if	NET-WORTH/SIZE = <0.2081
90	0.4	-0.4	HIGH if	EST-X-CHG-EPS-QTR-2 = >16.15
91	0.3	-0.3	HIGH if	FIN-STR-NUM = <5.5
92	0.3	-0.3	HIGH if	FLUCTUATION = 1.487<>2.108

Figure 7: Evidence for the return on investment prediction for STL

7 Related Work

A growing number of systems that perform database mining using artificial intelligence techniques have been developed in recent years. We compare *Recon* to the ones that have been most widely reported in literature. The comparison is done along two dimensions; the type of database mining operations supported by each system, and the degree of integration among each system's components. We compare *Recon* to four systems IMACS [1], KDW [7], INLEN [3], and MLT [9]. A more complete list of database mining systems, along with a more elaborate comparison methodology, are provided in [6].

Our first observation is that, with the exception of IMACS, previously reported systems support only a subset of the three operations we identified with database mining in Section 2. In particular, IMACS includes a deductive database processor to perform testing and refinement of rules hypothesized by the analyst, and a tightly integrated visualization component that is used to display the results of a query. *Recon*'s visualization component is not as tightly integrated with the other components. Data is communicated between the components through the *Recon* server. Another difference between IMACS and *Recon* is that the former does not interact with existing databases; data from such databases has to be loaded into memory before hypotheses can be tested. *Recon* interacts directly with relational database management systems through SQL. IMACS can be used to express and test more complex structures than *Recon*. This is because *Recon* is currently limited by what can be expressed in SQL.

From the pattern discovery perspective, *Recon* is most closely related to the KDW, INLEN, and MLT systems in three respects. First, all three systems use the same methods (inductive learning, and visualization) to perform pattern discovery. *Recon* currently includes a single rule induction system, whereas each of the other three systems includes a multitude of methods that provide the capability for broad experimentation. Such experimentation is necessary in research but not as much in application development. Second, all systems allow knowledge sharing between the components they integrate. The degree of component integration in these systems varies. *Recon* and KDW are loosely integrated systems. The components of INLEN and MLT are more tightly integrated. Each of these two systems uses a custom-developed knowledge representation that allows the components to share knowledge.

8 Conclusions

We have successfully applied *Recon* in: (1) financial domains to develop prediction models from data about stocks and commodities, (2) manufacturing domains to extract information that allowed for the correction of manufacturing process failures, and (3) demographic data to identify potential customers in product marketing campaigns. Our work to date has allowed us to reach the following conclusions:

1. Effective database mining is performed through the interweaving of several operations, the primary of which are testing and refining rules proposed by the analyst, and automatically discovering rules.

2. Two or more techniques often need to be used cooperatively to achieve the desired result. *Recon* provides the analyst with techniques for all identified database mining operations. These techniques have been integrated in a way that enables them to be used cooperatively.

Current work on *Recon* includes the improvement of the search techniques used by its artificial intelligence modules, improvement of the server's functionality, and the application of the system in additional domains.

References

[1] R. Brachman, P. Selfridge, L Terveen, B. Altman, F. Halper, T. Kirk, A. Lazar, D. McGuiness, L. Resnick, and A. Borgida. Integrated support for data archaelogy. In *Proceedings 1993 AAAI Workshop on Knowledge Discovery in Databases*, pages 197–211. AAAI, 1993.

[2] Usama Fayyad and Padhraic Smyth. Automated analysis of a large-scale sky survey: the skicat system. In *Proceedings of the Knowledge Discovery in Databases (KDD-93) Workshop*, pages 1–13, 1993.

[3] K. Kaufman, R. Michalski, and L. Kerschberg. Mining for knowledge in databases: Goals and general description of the INLEN system. In *Proceedings of the 1991 AAAI Workshop on Knowledge Discovery in Databases*, pages 35–51. AAAI, 1991.

[4] C. Kellogg and B. Livezey. Intelligent data exploration and analysis. In *Proceedings of the Conference on Information and Knowledge Management (CIKM-92)*, 1992.

[5] R. Kerber. Learning classification rules from examples. In *Proceedings 1991 AAAI Workshop on Knowledge Discovery in Databases.* AAAI, 1991.

[6] C. Matheus, P.K. Chan, and G. Piatetsky-Shapiro. Systems for knowledge discovery in databases. *IEEE Transactions on Knowledge and Data Engineering*, 5(6), 1993.

[7] G. Piatetsky-Shapiro and C. Matheus. Knowledge discovery workbench for exploring business databases. *Intarnational Journal of Intelligent Systems*, 7, 1992.

[8] E. Simoudis, D. Klumpar, and K. Anderson. Rapid visualization environment: Rave. In *Proceedings of the 9th Goddard Conference on Space Applications of Artificial Intelligence*, May 1994.

[9] Marc Uszynski. Machine learning toolbox. Technical report, European Economic Community, Esprit II, 1992.

[10] W. Ziarko, R. Golan, and D. Edwards. An application of datalogic/r knowledge discovery tool to identify strong predictive rules in stock market data. In *Proceedings of the Knowledge Discovery in Databases (KDD-93) Workshop*, pages 89–101, 1993.