Using Recon for Data Cleaning

Evangelos Simoudis*, Brian Livezey, and Randy Kerber
Lockheed Palo Alto Research Laboratories
Organization 96-20, Building 254G
3251 Hanover Street
Palo Alto, CA 94303-1191
{livezey,kerber}@aic.lockheed.com

Abstract

To aid in making investment decisions, financial analysts purchase large amounts of data from financial data providers. They use this historical data to develop financial models for making predictions and assessing risk about current dat about current dataa. Unfortunately, these database often contain errors and omissions of important information. Analysts are dependent upon the quality of these databases—missing data can prevent computation of key values and, more dangerously, incorrect data can cause their models to produce erroneous results without the analyst's knowledge. Because of the importance of accurate data, and the large volume of data involved, data providers and consumers have a need to develop advanced methods for *data cleaning*: the process of identifying and correcting incomplete and incorrect information in databases.

This paper describes how the *Recon* data mining system has been used to clean financial databases. Recon incorporates several data mining modules into a single, uniform framework: data visualization, deductive databases, and rule induction. The data visualization component supports the visual detection of outliers and other unusual phenomena. The deductive database enables analysts to capture, codify, and apply corporate knowledge about data integrity. The rule induction module creates error detection rules by generalizing from known errors to detect suspicious data entries in the rest of the data. The collaborative use of these three modules yields superior error detection over the application of any single data mining technique.

Key Words: data cleaning, data mining, deductive databases, rule induction, data visualization

*Evangelos Simoudis is now at IBM Almaden Research Center, 650 Harry Road, San Jose, CA 95120-6099.

Introduction

Advances in information technology have led to a rapid increase in the amount of data stored and disseminated electronically. The success of many enterprises is dependent upon their ability to accurately and rapidly make use of this information to reach effective decisions. In addition, the need to handle large amounts of data efficiently leads companies to develop and use software tools to automatically reason with the data.

The area of financial markets is, in particular, a data-intensive industry. A number of companies, such as Moody's, Standard and Poor, Bloomberg, and Dow Jones, are in the business of collecting, packaging, and selling data related to stock, bond, commodity, and currency markets. The information available ranges from historical data on prices and economic indicators, to information from corporate financial reports (e.g., assets, liabilities, sales), to real-time feeds that provide nearly instantaneous price updates.

A consequence of this increased reliance on electronic data is an increased vulnerability to errors in the data, especially when automated methods are employed. A simple typo, such as entering "38625" instead of "38.625", could cause a computerized trading system to make disasterous trades or a complex economic model to return very misleading results.

The large volume of data involved frequently makes human verification of accuracy impractical. As a result, it becomes necessary to implement computerized methods of examining databases, detecting missing and incorrect data, and correcting errors. This process is commonly known as *data cleaning*.

Recon (Kerber, Livezey & Simoudis 1995, Simoudis, Livezey, & Kerber 1994) is a multi-component data mining system that has been used to improve data quality. *Data mining* is the process of extracting previously unknown patterns from large databases. In the case of data cleaning, the patterns of interest are those that allow us to detect errors in the data.

We describe how *Recon* was used to clean data in two financial applications. The first application involves the development and validation of techniques for detecting errors in a real-time data feed. In the second application, *Recon* was used to develop models that would permit detection of errors and missing data in a static database of terms and conditions for government and corporate bonds.

Section describes the *Recon* data mining environment and describes briefly how each of its modules was used in the data cleaning tasks. Section describes the importance of data cleaning. Section describes the application of *Recon* in two financial data cleaning applications. Section briefly compares our work to other data cleaning work. The final section presents our conclusions about using *Recon* to identify errors in databases.

Recon

Recon is a multi-component data mining system that performs top-down and bottom-up data mining. Its architecture permits the cooperative application of multiple data mining techniques to a single data mining problem.

In top-down data mining, *Recon* assists analysts in expressing concepts, as well as in hypothesizing and testing patterns that may exist in the database being mined. For example, the analyst may define the concepts "clean bond," and "stale bond," and the pattern, "If a bond is stale then it is not clean." *Recon* uses a deductive database (Livezey & Simoudis 1995) for top-down data mining.

In bottom-up data mining, *Recon* automatically extracts rules from the contents of a database. *Recon* uses rule induction (Kerber 1991), conceptual clustering, neural networks, and nearest neighbor to perform bottom-up data mining. In addition, *Recon*'s interactive visualization component (Simoudis, Klumpar, & Anderson) can be used to visually explore data to identify patterns, outliers, and clusters of interest. However, for the results of the visual analysis to be useful, the user must carefully select the features of the data to be visualized. Finally, the user must manually encode the concepts that are inferred from a particular visualization.

Recon interfaces with several commercial relational database management systems, spreadsheets and flat files. The system runs under the UNIX operating system.

Data Cleaning

The quality of value-prediction models used by analysts is limited by the quality of the data from which it was created. In addition, even a perfect model cannot be expected to produce reliable results when the data it must use to make a prediction is noisy, incomplete, or incorrect. Despite the importance of accurate data, and the potentially serious consequences of relying on it, errors are quite common in the databases used by financial analysts. Consequently, financial data producers and consumers have a need for techniques that can identify erroneous entries in databases and correct or remove them. Unfortunately, existing data cleaning techniques are often inadequate for achieving the desired level of data quality.

These techniques suffer from three problems. First, they rely on an *ad hoc* methodology for setting and tuning data integrity constraints. Second, data integrity constraints must be expressed imperatively making it impossible for analysts to obtain explanations of why a particular record is considered suspicious. Finally, they offer no means for discovery of new types of errors and error detection techniques. *Recon* addresses these shortcomings by providing:

- a declarative language in which analysts can express and refine domain knowledge — explanations for constraint violations are provided in terms of this knowledge
- interactive visualizations that allow analysts to explore data visually in order to detect anomalies and trends
- a rule induction module that can automatically fine-tune existing integrity constraints and discover new types of errors

Using Recon to Clean Financial Data

In this section, we report the preliminary results of using *Recon* to clean financial data in two different applications. The first application involves the development of techniques for detecting errors in a data feed reporting financial transactions. The second application involves the development of models that permit detection of errors and missing values in a database containing information on terms and conditions of government and corporate bonds.

Transaction Reports

The first data set contained trading data about 4000 instruments collected over a 22-day period from the Frankfurt Stock Exchange. It contained four types of data: trading data about each financial instrument traded as it is reported by the exchange, data about the same financial instruments and trading actions as reported by an independent source, the closing price of each financial instrument on the trading day before the start of the 22-day period, and the notifications for suspicious transactions detected by the customer through the use of traditional error detection techniques and manual investigation. The data was stored in an Oracle DBMS.

After examining the provided data and consulting with data analysts, we decided to use *Recon* to identify two types of suspicious transactions in the provided data: 1) historic high/lows, and 2) large price changes. A Historic High/Low event occurs when an instrument's feed price exceeds its previous high or falls below its previous low. A Large Price Change event occurs when an instrument's price changes by more than some specified relative or absolute threshold. Included in the record for each Large Price Change event is the instrument's new price, the absolute and relative changes, and the type of price quote (i.e., open, trade, or close).

The integrity tests obtained from the customer yielded far too many *false positives* : transactions that were flagged as errors but were actually correct. We used data visualization and rule induction to refine some of the thresholds in these integrity checks. The goal of the refinement process was to reduce the number of false positives while still detecting a significant number of actual errors.

Figure 1 shows a two-dimensional scatter plot of *absolute* versus *relative* price difference (with respect to the previous high/low) for all the historic high/low events detected in the provided database. The events that were confirmed as errors are shown as squares; all others are shown as plus signs. The upper window shows the entire space, while the lower window gives a more detailed view of the region bounded by the small box in the lower left corner of the upper window. By examining this visualization of the data and testing and refining hypotheses within *Recon*'s deductive database, we arrived at the following restrictions on the parameters for a Historic High/Low event:

$$\begin{array}{llr} absolute > 2.0 \ \ and \ \ relative > 0.8 & & (1) \\ relative > 1.0 & & (2) \\ absolute > 2.0 \ \ and \ \ relative > 0.4 & & \\ \qquad\qquad\qquad and \ \ direction = low & & (3) \end{array}$$

If any of these three conditions hold, then the event is considered an error.

For the Large Price Change events, *Recon*'s visualization module was first used to identify *outliers*, those transactions that are radically different from the others. From this examination, the following parameter restrictions were inferred:

$$\begin{array}{lr} relative > 20.0 & (4) \\ price > 1000 \ \ and \ \ relative > 5.0 & (5) \end{array}$$

If either of these conditions holds, then the event is considered an error. These thresholds were determined by examining interactive visualizations similar to the one in Figure 1.

Figure 2 shows a two-dimensional scatter plot of *absolute* versus *price* for all Large Price Change events. The lower window shows more detail for events where *absolute* $<$ 300 and *price* $<$ 300. Two aspects of this visualization are very important. First, note the dark band in the lower window (the plot symbol is a diamond[1]). Each of the events in this band involved the same instrument (IC = 569213). Most of the bands seen in the visualization contain events involving a single instrument — *this suggests that instrument-specific thresholds might be warranted.* The second important aspect of Figure 2 is that it allows us to isolate a relatively small subset of the data that includes a relatively large number of confirmed errors. With this smaller subset of data, we can use Recon's Rule Induction module to automatically induce appropriate parameter thresholds. These thresholds will allow us to identify Large Price Change events that are actually errors without including too many false positives. While visualization is inherently limited in the number of variables in any derived restriction, rule induction can consider arbitrarily many variables. It is, however, self-limiting in order to prevent *over-fitting* of the data.

[1]These visualizations are much more compelling when color can be used for discrimination

The *Recon* rule induction module yielded the following restrictions:

$$\begin{array}{lllr} absolute > 300 & and & price > 300 & \\ & and & relative > 2.5 & \\ & and & absolute > 2.0 & (6) \\ absolute > 300 & and & price > 300 & \\ & and & type = trade & \\ & and & emsn < 1039600 & (7) \end{array}$$

These restrictions are combined with restrictions 4 and 5 above. If any of the four restrictions hold, the event is considered to be an error.

Table 1 indicates how many errors the refined event checks successfully identified, as well as the number of false positives the event check generated. As a point of reference, the threshold tests being used currently detected a total of 10 errors with 413 false positives.

Event Type	Number Confirmed as Errors	False Positives
Historic High/Low	9	0
Large Price Change	5	13

Table 1: Error detection rate and false positive rate after refinement

The application and testing of hypothesized error-detection knowledge, along with the discovery of new knowledge, yielded very promising results. The error detection rate increased while the false positive rate simultaneously decreased dramatically. *Recon*'s deductive database, visualization, and rule induction modules proved essential in achieving these results.

The discovery that certain instruments consistently exhibited very large price changes might allow us to exploit instrument-specific thresholds when checking for large price changes. By excluding such instruments, the false positive rate would be reduced dramatically, allowing us to loosen some of the constraints and thereby detect more true errors. Simple volatility measures, where an instrument's feed price is compared to the previous day's high and low, might also yield further error detections.

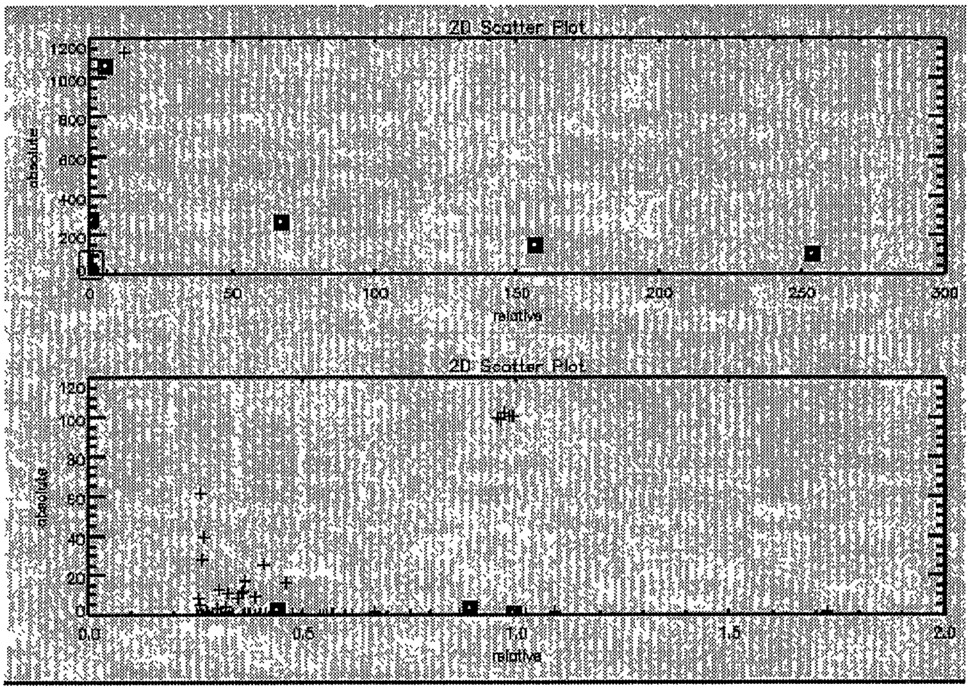

Figure 1: Historic high/low events

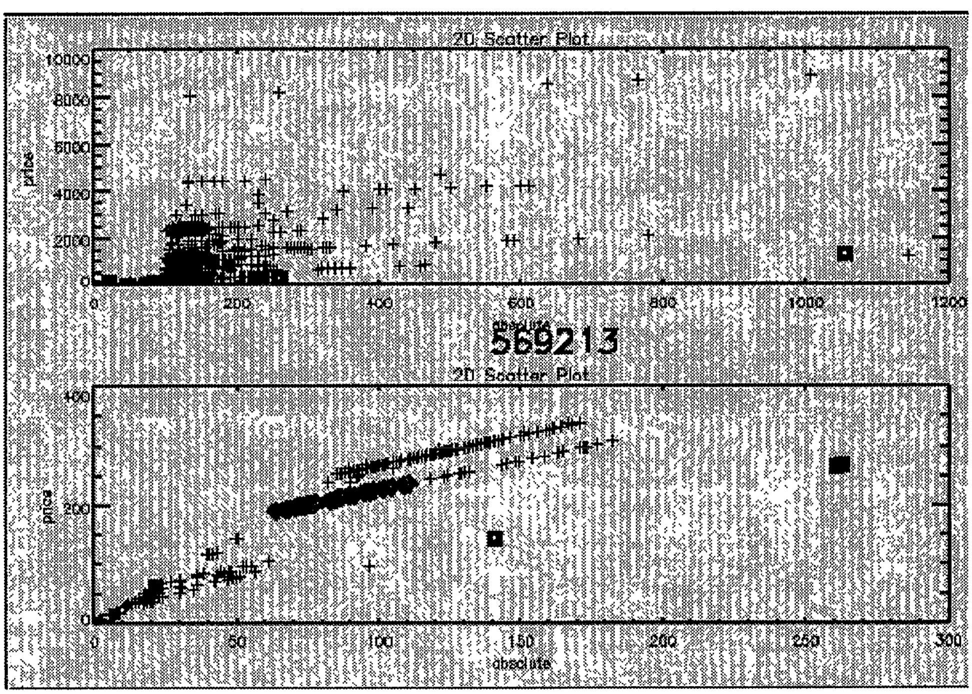

Figure 2: Large Price Change events

With more data, we could have further exploited the Rule Induction module to optimize the thresholds for each of the event types detected. Furthermore, each individual test could contribute a confidence factor to how likely a given event is to be an error. Since so few known errors were provided, such activities would likely lead to over-fitting of the data in this case.

Bond Data: Terms and Conditions

The other data cleaning operation involved a database of 2200 Mexican and British government bonds and Eurobonds. The database consisted of 10 tables and about 150 fields describing terms and conditions and background information about each bond.

Traders and analysts use this data to build bond portfolios and develop and apply valuation models of fixed income financial instruments. These models help analysts evaluate the rate of return and risk of bonds under various scenarios of future interest, inflation, tax, and currency exchange rates. If certain vital information is missing, it might not be possible to perform the desired calculations. Errors in the data can lead to errors in the constructed models and erroneous results when the models are applied to specific bonds, possibly leading to disasterous trading decisions.

This bond database did not previously have rigorous cleaning mechanisms in place, except for a database administrator occasionally issueing queries to check for certain well-defined domain value violations. The first step in the data cleaning process was a series of knowledge acquisition sessions with bond analysts, during which we obtained a list of integrity violations that they remembered having encountered in the past. After each session, we encoded these constraints in the deductive database module and ran them against the bond database. Based on analyst feedback, we further refined the concepts and added our own based on anomalies that we noticed. For example, after first encoding the constraint $issue\_date < maturity\_date$, the bonds returned included a number for which $issue\_date = 0$ (which is an error). This resulted in constructed a new concept to check for $issue\_date = 0$ and refining the other concept to be $issue\_date > 0$ *and* $issue\_date < maturity\_date$.

Recon's visualization component was used to detect outliers, which often correspond to errors. For example, when looking at a distribution of the values for the field *coupon*, there were a large number of points at a value of around 100 and no points between 30 and 100. Via further investigation with the deductive database component, we found that there were approximately 100 bonds whose value for *coupon* was set to 99.99. By plotting *coupon* versus $maturity\_date$, we found that most of these bonds were rather old. After consulting with the bond analysts, we were told that many of these bonds should have been removed from the database as they had expired, and that the value of 99.99 was sometimes used for *coupon* when the data entry person was unsure what value to enter.

The final data integrity model included over 50 concepts. Examples of concepts in the model include:

- **Domain Value.** Flag fields should have a value of "Y" or "N".
- **Missing Reference.** If $floating\_flag = Y$ then there should be an entry for this bond in the floating_formula table.
- **Duplication.** If two records have the same value for *company_name*, $issue\_date$, $maturity\_date$, *coupon*, and *redemption_value* then they are likely to be the same bond entered twice.

Related Work

Knowledge-based error detection of the type reported in this paper has only recently started to attract the attention of data mining researchers. In particular, the Q-Data system (Sheth, Wood, & Kashyap 1995), which is based on the LDL++ deductive database (Tsur, Arni, & Ong 1992, Zaniolo 1992), allows a user to express tentative knowledge. It is similar in operation to *Recon*'s deductive database. However, Q-Data does not use induction and visualization techniques which complement and augment the error-detection capabilities of a deductive database.

Guyon *et al* in (Guyon, Matic, & Vapnik) describe a purely inductive approach to identifying erroneous data. In particular, they first examine the characteristics of each extracted pattern, e.g., the pattern's information gain, to identify suspicious patterns, e.g., patterns with very high information gain, and then remove such patterns as well as the data that gave rise to them. The major drawback of this approach is that with very large databases, where the number of discovered patterns is very large, it will be difficult for a user to select the set of patterns to remove. The balanced approach of *Recon* which combines top-down with bottom-up data mining allows the analyst to better focus the search for suspicious database records.

Errors can also be detected using statistical techniques by identifying deviations from certain norms that can be established in the data. Such an approach is taken by the KEFIR system (Piatetsky-Shapiro & Matheus 1994), although the system has not been used for error detection. The major advantage of the approach taken by *Recon* is that it provides the user with an explanatory capability; i.e., the analyst can ask the system for an explanation of why a particular record was considered an error. KEFIR uses hand-crafted knowledge that can be used to explain certain types of pre-defined deviation types.

Conclusions

We have reached three conclusions from our work thus far.

1. The development of sophisticated models employed by decision makers to reach complex decisions is inhibited by the low quality of the available data. Existing data cleaning techniques are inadequate for correcting the contents of existing databases.
2. Analyst knowledge alone is inadequate for detecting errors in large financial databases. Additional knowledge needs to be expressed, tested, and used to explain test results. This knowledge must also be complemented with symbolic error detection knowledge that is discovered in the database. Over time tentative knowledge, hypothesized and discovered, becomes corporate knowledge and is permanently encoded in the database.
3. Deductive databases, rule induction, and visualization can be used cooperatively to express and discover tentative error-detection knowledge.

References

Guyon, I.; Matic, N.; and Vapnik, V. 1994. Discovering Informative Patterns and Data Cleaning. In Proceedings 1994 AAAI Workshop on Knowledge Discovery in Databases. 145-156. Menlo Park, Calif.:AAAI.

Kerber, R. 1991. Learning Classification Rules from Examples. In Proceedings 1991 AAAI Workshop on Knowledge Discovery in Databases. Menlo Park, Calif.:AAAI.

Kerber, R.; Livezey, B.; and Simoudis E. 1995. A Hybrid System for Data Mining. Goonatilake, S., and Khebbal, S. eds. *Intelligent Hybrid Systems.* New York, New York: John Wiley and Sons.

Livezey, B., and Simoudis, E. 1995. A Deductive Front-End for Relational Databases. Ramakrishnan, ed. *Applications of Logic Databases.* 57-81. Boston, Mass.:Kluwer Academic Publishers.

Piatetsky-Shapiro, G., and Matheus, C. 1994. The Interestingness of Deviations. In Proceedings 1994 AAAI Workshop on Knowledge Discovery in Databases. 145-156. Menlo Park, Calif.:AAAI.

Sheth, S.; Wood, C.; and Kashyap, V. 1995. Q-Data: Using Deductive Database Technology to Improve Data Quality. Ramakrishnan, ed. *Applications of Logic Databases.* 23-56. Boston, Mass.:Kluwer Academic Publishers.

Simoudis, E.; Klumpar, D.; and Anderson, K. 1994. Rapid Visualization Environment: RAVE. In Proceedings of the 9th Goddard Conference on Space Applications of Artificial Intelligence.

Simoudis, E; Livezey, B.; and Kerber, R. 1994. Integrating Inductive and Deductive Reasoning for Database Mining. In Proceedings of the 1994 Workshop on Knowledge Discovery in Databases. 37-48. Menlo Park, Calif.:AAAI.

Tsur; Arni; and Ong 1992. The LDL++ User Guide. Technical Report Carnot-012-93(P). MCC.

Zaniolo, C. 1992. Intelligent Databases: Old Challenges and New Opportunities. *Journal of Intelligent Information Systems*, 1:271-292.