Data Quality Models for High Volume Transaction Streams: A Case Study

Joseph Bugajski
Visa International
Foster City, CA USA
JBugajsk@visa.com

Robert L. Grossman
Open Data Group
River Forest IL USA
& National Center for Data Mining
University of Illinois at Chicago
Chicago IL USA
rlg1@opendatagroup.com

Chris Curry, David Locke & Steve Vejcik
Open Data Group
River Forest IL USA
{ccurry, dlocke, vejcik}@opendatagraoup.com

ABSTRACT

An important problem in data mining is detecting significant and actionable changes in large, complex data sets. Although there are a variety of change detection algorithms that have been developed, in practice it can be a problem to scale these algorithms to large data sets due to the heterogeneity of the data. In this paper, we describe a case study involving payment card data in which we built and monitored a separate change detection model for each cell in a multi-dimensional data cube. We describe a system that has been in operation for the past two years that builds and monitors over 15,000 separate baseline models and the process that is used for generating and investigating alerts using these baselines

Categories and Subject Descriptors

G.3 [Probability and Statistics]: Statistical computing, statistical software

I.5.1 [Models]: Statistical Models

General Terms

Algorithms

Keywords

baselines, data quality, change detection, cubes of models

1. INTRODUCTION

It is an open and fundamental research problem to detect interesting and actionable changes in large, complex data sets. In this paper, we describe our experiences and the lessons learned over the past three years developing and operating a change detection system designed to identify data quality and interoperability problems for Visa International Service Association ("Visa"). The change detection system produces alerts that are further investigated by analysts.

The problem is difficult because of the following challenges:

1. Visa's data is high volume, heterogeneous and time varying. There are 6,800 payment transactions per second that must be monitored from millions of merchants located around the world that are processed over a payment network that connect over 20,000 member banks. There are significantly different patterns across regions, across merchants, during holidays and weekends, and for different types of cardholders. See Figure 1 for example.

2. Alerts arising from the change detection system generally require human examination. Because of this it is necessary to balance generating a meaningful number of alerts versus generating a manageable number of alerts. If too many alerts are generated, it is not practical to manage them. If too few alerts are generated, they are generally not meaningful.

In this paper, we describe our experiences using a methodology for addressing these challenges. The methodology we use is to build a very large number of very fine grained baselines, one for each cell in a multi-dimensional data cube. We call this approach Change Detection using Cubes of Models or CDCM.

For example, we built separate baseline models for each different type of merchant, for each different member bank, for each different field value, etc. In total, over 15,000 different baseline statistical models are monitored each month and used to generate alerts that are then investigated by analysts.

We believe that this paper makes the following contributions:

1. First, we have highlighted an important category of data mining problems that has not received adequate coverage within the data mining community and whose importance will continue to grow over time.

2. Second, we have introduced a new change detection algorithm called CDCM that is designed to scale to large, complex data sets by building a separate change detection model for each cell in a data cube.

3. Third, we have introduced a software architecture that can reliably scale with tens of thousands of different individual statistical or data mining models.

4. Fourth, we have described how we dealt with some of the practical challenges that arise when deploying data mining and statistical models in operation.

Section 2 contains some background on payment card transactions. Section 3 describes the Change Detection using Cubes of Models (CDCM) algorithm that we introduce. Section 4 describes some typical alerts detected by the system we developed and deployed. Section 5 describes the program structure that we developed around the baseline models and monitor. Section 6 describes some of the issues arising when validating the models we developed. Section 7 describes the architecture of the system we developed. Section 8 describes the implementation. Section 9 describes some of the lessons learned. Section 10 describes related work. Section 11 is the summary and conclusion.

2. BACKGROUND ON PAYMENT CARD TRANSACTIONS

2.1 Processing a Transactions

We begin by providing some background on payment card transactions that will make this paper more self-contained. In this section, we define cardholder, merchant, acquiring bank, and issuing bank and describe the major steps involved when using a payment card.

1. A transaction begins when cardholder purchases an item at a merchant using a payment card. The payment card contains an account number that identifies the cardholder.

2. The merchant has a relationship with a bank called the acquiring bank, which agrees to process the payment card transactions for the merchant. The acquiring bank provides the merchant with a terminal or other system to accept the transaction and to process it.

3. The acquiring bank has a relation with a financial payment system, such as those operated by Visa and MasterCard. The transaction is processed by the acquiring bank and passed to the payment system. Visa operates a payment system called VisaNet.

4. The payment system processes the transaction and passes the transaction to the bank (the issuing bank) that issued the payment card to the cardholder. In other words, one of the essential roles of the payment system is to act as a hub or intermediary between the acquirer and the issuer.

5. The issuing bank processes the transaction and determines if there are sufficient funds for the purchase, if the card is valid, etc. If so, the transaction is authorized; the transaction can also be declined, or a message returned asking for additional information. The issuing bank also has a relationship with the cardholder or account holder. For example, with a credit card, the cardholder is periodically billed and with a debit card the appropriate account is debited. For the year ending March 31, 2007, there were over 1.59 billion Visa cards in circulation.

6. For each of these cases, the path is then reversed and the transaction is passed from the issuing bank to the payment system, from the payment system to the acquiring bank, and from the acquiring bank to the merchant.

Our problem was to use baselines and change detection algorithms to help detect data and interoperability problems at Visa [6]. Payment data arrives at Visa from millions of merchant locations worldwide. For the year ending March 31, 2007, total annual global card sales volume was over USD $4.8 trillion[1]. Payment data is processed through risk management rules set by over 20,000 individual member banks (issuing and acquiring banks). These rules determine if a payment authorization request from a merchant either is approved or rejected by the paying bank. For this problem, we built separate baselines for a variety of data fields, for each member bank, and for thousands of merchants. Overall, over 15,000 separate baselines are currently used each month to monitor payment card transactions at Visa.

2.2 Baselines for Field Values

Note: The examples in this section are hypothetical and only used for the purposes of illustrating how to define baselines.

A payment card transaction typically includes a number of fields, such as information about the point of services (POS) environment, the merchant's type of business and location, the cardholder's identity, the transaction currency, the transaction amount, and bank routing information.

We begin with an informal description of baselines based upon a simplified example. In this simplified example, assume that one of the fields of interest describes characteristics of the point of service (POS). Specifically, we assume that this field can take the following (hypothetical) values: 00, 01, 02, 03, and 04.

For an observation period of a week, assume that the frequency of these values for a certain acquirer is given by the left handt table in Table 1. Later, during the monitoring, assume that distribution is instead given by the right hand table in Table 1. The observed distribution in Table 1 is similar, except the value 04 is six times more likely in the observed distribution compared to the baseline distribution, although in both cases the values 02, 03 and 04 still as a whole contribute less than 3% of the distribution.

The challenge for detecting significant changes is that the distributions depend upon many factors, including the region, the season, the specific merchant, and the specific issuer and acquirer.

[1] Data reflects all Visa programs except Interlink, PLUS, and commercial funds transfers in China as reported by member financial institutions globally and therefore may be subject to change.

Value	Percent
00	76.94
01	21.60
02	0.99
03	0.27
04	0.20
Total	100.00

Value	Percent
00	76.94
01	21.67
02	0.90
03	0.25
04	1.24
Total	100.00

Table 1. The distribution on the left is the baseline distribution. The distribution on the right is the observed distribution. In this example, the value 04 is over 6x more likely in the observed distribution, although the two dominant values 00 and 01 still account for over 97% of the distribution.

3. CHANGE DETECTION USING CUBES OF MODELS (CDCM)

In this section, we describe a methodology called Change Detection using Cubes of Models or CDCM that is designed to detect changes in large, complex data sets.

3.1 Change Detection Models

Change detection models are a standard approach for detecting deviations from baselines [3].

We first describe the cumulative sum or CUSUM change detection algorithm [3]. We assume that we know the mean and variance of a distribution representing normal behavior, as well as the mean and variance of another distribution representing behavior that is not normal.

More explicitly, assume that we have two Gaussian distributions with mean m_i and variance s^2_i for i equals 0 and 1

$$f_i(x) = \frac{1}{\sqrt{2\pi\mu_i}} \exp\frac{-(x-\mu_i)^2}{2\sigma_i}$$

The log odds ration is then given by

$$g(x) = \log\frac{f_1(x)}{f_0(x)}$$

and we can define a CUSUM score Z_n as follows [3]:

$$Z_0 = 0$$

$$Z_n = \max\{0, Z_{n-1} + g(x_n)\}$$

An alert is issued where the score Z_n exceeds a threshold.

Quite often the statistical distribution of the anomalous distribution is not known. In this case, if the change is reflected in the mean of the observations and the standard deviation is the same pre- and post-change, the generalized likelihood ratio or GLR algorithm can be used [3]:

$$G_k = \frac{1}{2\sigma^2}\max_{1\le j\le k}\frac{1}{k-j+1}\left[\sum_{i=j}^{k}(x_i-\mu_0)\right]^2, \qquad k>1$$

where m_0 is the mean of the normal distribution and s is the standard deviation of the both the normal and abnormal distributions, which are assumed to be Gaussian. Here k is fixed and determines the size of the window used to compute the score. Again, the detection procedure is to announced a change at the first up-crossing of a threshold by the GLR score.

3.2 Cubes of Models

The basic idea of the CDCM algorithm is that for each cell in a multi-dimensional data cube, we estimate a separate change detection model.

For the purposes here, we can define a *data cube* as usual, namely a multi-dimensional representation of data in which the cells contain measures (or facts) and the edges represent data *dimensions* which are used for reporting the data.

We define a *cube of models* as a data cube in which each cell is associated with a baseline model. See Figure 2.

3.3 Learning and Scoring Change Detection Models

In our model, we assume that there are a stream of events, which in our case are transactions, and each event can be assigned to one or more cells in cube of models. For example, for each project, each transaction is assigned to the appropriate cell(s), as determined by one of six regions, by one of over 800 Merchant Category Codes (MCCs), by one of 8 terminal types, etc. We also assume that various derived data attributes are computed from the event data to form feature vectors, which are sometimes called profiles in this context. The profiles contain state information and derived data and are used as inputs to the models as usual.

Estimating baseline models. To learn the baseline models, we take a collection of event data and process it as follows.

1. First, we assign each event to one or more cells in the data cube as appropriate.
2. Second, we transform and aggregate the events and compute the required profiles for each cell using the event data.
3. Third, for each cell, we use the resulting profiles to estimate the parameters for the baseline model, and output the baseline model.

Scoring baseline models. To score a stream of event data, we proceed as follows.

1. First, we retrieve the appropriate XML file describing the segmentation.

2. Next, we assign each event to one or more cells in the data cube as appropriate.
3. We then access the profile associated with each cell, and update the profiles using the new event.
4. We then use the profile as the input to the appropriate baseline model and compute a score.
5. Next, we process the resulting score using the appropriate rules to determine whether an alert should be produced.
6. Next, we apply XSLT transformations to the score to produce a report describing the alert.
7. Finally, if an alert is produced, we pass the alert to the required application or analyst.

4. SOME TYPICAL ALERTS

4.1 Summary

Since the data interoperability program began approximately three years ago, Visa and its trading partners have fixed 70 data interoperability issues. An improvement in annual card sales volume realized thereby is about $2 billion, compared with global annual card sales volume of $4.8 trillion. These “fixed Alerts” comprise 25% of data interoperability issues presently being investigated.

In this section, we describe four typical alerts that have been generated by the CDCM system. Currently, we compute alerts each month and re-estimate baselines several times a year.

It is important to remember when reading the case studies in this section that the issues identified by these alerts represent both a very small fraction of the transactions and a very small fraction of the total purchase dollars.

4.2 Dimensions of Cube

For the Alerts that we describe below, we used the following dimensions to define a data cube:

1. The geographical region, specifically the US, Canada, Europe, Latin America, Asia Pacific, Middle East/Africa, and other.
2. The field value or combination of values being monitored.
3. The time period, for example monthly, weekly, daily, hourly, etc.
4. The type of baseline report, for example a report focused on declines or a report describing the mixture of business for a merchant.

Today (June, 2007), for each of 324 field values times 7 regions times 1 time period times 3 report types, we estimate a separate baseline, which gives 324 x 7 x 1 x 3 = 6816. In addition, for 623 field values times 7 regions times 1 time period times 2 report types, we estimate a separate baseline, which gives an additional 623 x 7 x1 x 2 = 8726 separate baseline models. So in total, we are currently estimating 15,542 (=6816+876).

Actually, the description above is a simplified version of what actually takes place. For example, the 6816 baselines mentioned arise from 324 x 7 = 2272 different field values, but the 2272 different field values are not spread uniformly across the 7 regions as indicated, although the total is correct.

4.3 Incorrect Merchant Category Code

In this example, an airline was coding some of its transactions using a Merchant Category Code (MCC) B instead of the preferred MCC A, which coincided with a lower approval rate for the airline’s payment authorization requests. Lower authorization approval rates have been shown in work on other alerts to be associated with a loss of purchase value for Visa and its Member banks. These factors led to the production of a baseline alert that was followed by an analyst’s investigation. Once the analyst confirmed the issue, a conference call was arranged with a person responsible for the relationship with the acquiring bank for the airline. As a result of this call, the airline installed a fix that lead to improved authorization approval performance and increased annual purchase volume.

4.4 Testing of Counterfeit Cards

In this example, the decline rate for a large bank was essentially the same month to month but the baseline model identified a particular category of transactions (specified by a combination of five fields) for which the decline rate sharply peaked in September 2006 compared to an earlier baseline period. One way of thinking about this, is that for this bank, most of the 50,000+ or so baselines were normal for September, but one was not. When investigated, this particular baseline was elevated due to unauthorized testing of Visa accounts, a practice that sometimes is associated with underlying criminal activity; e.g., validation of active card accounts using illegally obtained card account data. Visa and the bank moved swiftly correct the problem following further investigation.

4.5 Incorrect Use of Merchant City Name

In this example, a European merchant’s transactions were coded incorrectly; i.e., incorrect information was contained in the data field that is used to encode the name of the city where the transaction occurred. This also was associated with a lower approval rate for payments from this merchant. The lower approval rate was detected by a baseline alert that monitored decline levels for each MCC for each acquirer. After investigation and communications with the acquirer, the merchant corrected the problem.

4.6 Incorrect Coding of Recurring Payments

A recurring payment is an arrangement agreed between a cardholder and merchant whereby a merchant periodically submits payment authorization requests on behalf of their customer for continual use of a product or service. Examples include monthly payments for mobile phones, internet, or satellite television services. Recurring payments are coded specially to indicate that the cardholder previously requested this Visa service of the merchant. In this case, a Middle Eastern merchant incorrectly coded payments as recurring and the decline rate was higher than it should have been had the transaction been coded otherwise. In addition, after examination by an analyst, it became clear that the merchant name was also inconsistently coded, compounding the problem. This alert was detected using a MCC-baseline.

5. PROGRAM STRUCTURE

In this section, we describe the structure of the program created to monitor and improve data quality. See Figure 3.

Strategic Objective. Critical to the success of the program was identifying the strategic objective of the program. After much discussion, the following objective was agreed to: identify and ameliorate data quality and data interoperability issues in order to maintain and improve 1) the approval of valid transactions; 2) the disapproval of invalid or fraudulent transactions; and 3) the correct coding of transactional data. The first two objectives increase the satisfaction of card holders and member banks, while the third objective lowers the overall cost and increases the efficiency of transaction processing. The success of the program was then tracked by a dashboard that monitored the additional dollars processed and the savings resulting from direct actions of the program.

Governance. The Visa Data Interoperability Program was established in 2004 by the global council of the CIO's of Visa's operating units. The program is governed by a council of business executives and technical experts who set the rules, procedures and processes of the program.

Monitoring. A system supporting data cubes of baseline models was designed and developed using the ideas described above to monitor transactional payment data. For the purposes here, we call the system the Monitor. The Monitor receives daily samples of tens of millions of authorization messages and clearing transactions from a central ETL facility inside VisaNet. Statistically significant deviations from baselines that are associated with high business value generate what are called Baseline Threshold Alerts.

Screening of Alerts. Baseline Threshold Alerts are screened by an analyst to produce what are called Baseline Candidate Alerts. Candidate alerts are then analyzed by program analysts and other subject matter experts to understand the issues that led to the candidate alert and to more carefully estimate the business value involved. If the program team believes that an issue is valid and sufficiently valuable that the cost of repair may be recovered through recaptured revenue or lower processing costs, and, furthermore, they believe that the issue is sufficiently clear that it may be explained accurately, they send a Program Alert to the customer relationship manager at the Visa operating region that is closest to the source of the problem. This may be a third party processor, an acquiring bank, or a VisaNet technical group.

Investigations. The customer relationship manager works with the program analysts to explain the problem identified by the Program Alert to the bank or merchant and to work with them to estimate the cost required to fix the problem. The program team meanwhile reviews measurements to determine when and if the problem is resolved. If the data measurements indicate a resolution, then the business that effected the change is contacted once again to validate recovery of revenue or loss avoidance.

Reference Model. The governance council adopted technical standards for how alerts are generated and business process standards for how alerts are investigated. These rules are recorded in a Reference Model that is maintained by the Program and updated at least twice each year.

Standards. Over time, we developed an XML representation for baseline models and for segmentation. We worked with the PMML Working Group and this work has not contributed to the PMML Baseline and Change Detection Model, which is currently in a RFC status. Over the long term, this should reduce the total costs of the system by enabling the use of third party tools that support this RFC draft standard.

6. VALIDATION

Fraud models are relatively easy to validate using detection rates and false positive rates. Response models are relatively easy to validate using lift curves.

On the other hand, the change detection models we used are somewhat harder to validate. Broadly speaking, change detection models are similar to association rules in that for a given threshold and support, an association algorithm will find all corresponding association rules. Of the ones found, some may have business significance, while others may not. An analyst is needed to tell the difference.

The change detection models we used for this project are similar in the sense that for a given segmentation and threshold, all corresponding changes are detected by the algorithm, but not all have business significance. Again, an analyst is needed to separate those with business significance from those without. The role of the baseline models is simply to produce a useful flow of alerts for investigation. The segmentation process is critical so that the alerts are both meaningful (the segment is no so broad that the alerts are without business significance) and manageable (the segment is not too narrow, producing too many alerts).

As the program matured, we have introduced several rules that have resulted in more useful alerts. First, as described above, we introduced a quick manual screening process before Baseline Candidate Alerts are generated. Second, we have introduced various thresholds, which depend upon the issue, region, and other factors, involving number of transactions, the type of transactions, and the dollar amount, etc. that are used in part to determine Baseline Candidate Alerts.

7. SYSTEM ARCHITECTURE

In this section, we describe the system architecture that we developed for this project. See Figure 4. We believe that this architecture is quite general and would prove useful for a variety of projects.

7.1 Predictive Model Markup Language

Metadata for the system was stored using the Predictive Model Markup Language or PMML [5]. PMML was used to specify the data attributes (using the PMML data dictionary), the attributes used as inputs for each model (using PMML mining attributes), the specification for derived functions (using the PMML transformation dictionary and local transformations), and the specifications of model segments (using a PMML extension we helped define).

Additional metadata about the system was stored in an emerging part of PMML called the PMML deployment package. Using the PMML deployment package, we specified such things as the source and location of various data feeds, as well as how the data should be pre-processed and post-processed.

7.2 Baseline Producers and Consumers

The main two system components are a baseline producer and a baseline consumer.

The baseline producer extracts transactions data from a project data mart, computes derived attributes and state information from the transactions (which are sometimes called profiles), and then uses the information to estimate the parameters of a separate baseline model for each segment. The baseline models for each segment are saved as an PMML.

The baseline consumer first reads a PMML file specifying the segmentation and the model parameters for each segment. The baseline consumer than processes a stream of transaction data and produces a stream of scores. Sometimes a PMML consumer is called a scoring engine since it processes input data using a PMML file to produce a stream of output scores.

7.3 XSLT-based Report Processing

An important part of the architecture turned out to be a basic module that took XML files of scores produced by the baseline Consumer and transformed them using definitions for reports specified using XSLT. In this way, we could relatively easily change the report formats. Being able to experiment quickly and easily with different report formats turning out to be an important for the project, as we describe in more detail below in the section on lessons learned.

7.4 Metadata Repository

We stored the various metadata for the project, including the PMML files, in a repository. This was important since it was a project requirement that we be able to retrieve the specification for any baseline that had been computed by the project. Using the repository, which was backed up automatically, also satisfied a business continuity requirement of the project.

7.5 Defining Cubes of Models Using Segmentation

As we mentioned, the success of the approach was critically dependent on being able to define and manage easily ten of thousands of different segments. To do this we developed a PMML extension to define how data should be divided in order to define different segments, each of which was used to estimate a separate model.

After several refinements over the past two years, we defined several different ways to define segments. For this project, we mainly used the following mechanisms for defining segments:

- Regular Partitions. With a regular partition, a field name, the left end point, the right end point, and the number of partitions is specified. Regular partitions in two or more dimensions can be defined by specifying the required data for each field independently.
- Explicit Partitions. With an explicit partition, the field name, the left end point, and the right end point are given for each interval in the partition. Note that with explicit partitions, the intervals may be overlapping. Again, multi-dimensional partitions are defined by defining each dimension independently.
- Implicit Partitions. With an implicit partition, a field name is provided and then each unique value of the field is used to define a distinct partition. For example, assume that city is a field in a data set that is identified as an implicit partition field. In this case, a separate model would be created for each city.

8. IMPLEMENTATION

8.1 Augustus

In part motivated by this project, we developed an open source PMML-compliant data mining system called Augustus. Augustus, which is available through Source Forge (www.sourceforge.net/projects/augustus), includes a baseline producer and a baseline consumer.

Augustus is written in Python. The current version of Augustus is 0.2.6. The Augustus kernel contains a data management component called UniTable for Universal Table. A UniTable is broadly similar to a R data frame. Data is arranged in columns, the columns may be of different types, but all columns must have the same number of rows.

Derived columns can be easily added to a UniTable and many columns operations can be vectorized. UniTable is based on the Python numarray library.

Augustus includes a library for working with PMML and also a library for importing data. Augustus currently is distributed with two models: a baseline model and a tree mode

8.2 Scalability of Augustus

Transactions on VisaNet can peak at over 6,800 transactions per second. For this reason, an important requirement of this project is that our Augustus-based scoring must also be able to process events at this speed. Table 1 are some sample benchmarks for scoring showing that scoring is independent of the number of events scored, which is an important requirement for our project.

Another important requirement for our implementation is that scoring of events and building of baselines be independent of the number of segments. The same table shows that Augustus satisfies this requirement also.

Number of Events	**Number of Explicit Segments**	**Events/sec**
1,000,000	10	14,880
1,000,000	100	15,216
1,000,000	1,000	14,808
1,000,000	10,000	13,790
2,000,000	10	15,100
2,000,000	100	15,278
2,000,000	1,000	14,401
2,000,000	10,000	14,320

4,000,000	10	15,198
4,000,000	100	15,251
4,000,000	1,000	15,137
4,000,000	10,000	14,845
8,000,000	10	15,299
8,000,000	100	15,407
8,000,000	1,000	15,367
8,000,000	10,000	14,745

Table 2. The events per second processed by the Augustus scoring engine is approximately independent of the number of events scores and the number of segments.

8.3 PMML Baseline Models

Over the past few years, we have worked with the PMML Working Group to develop a PMML representation for baseline models, which was developed in part to support this project. This proposal has been modified by the PMML Working Group and the modified proposal is currently available as a PMML RFC (Request For Comment) [5]. The proposal for baseline models includes CUSUM models, GLR models, threshold break models, (a measure exceeds a threshold), and contingency table models.

We also developed, in part motivated by this project, a PMML extension for defining segmented models. Some of the different types of segments supported are described above. Having a easy to define segments turned out to be very useful for this project.

9. DISCUSSION AND LESSONS LEARNED

Lesson 1. The most important lesson we learned was that thus far it has been more fruitful to examine many individual baseline and change detection models, one for each different segment of the event stream, even if the these models are very simple, than to build a single, relatively complex model and apply it to the entire event stream.

Lesson 2. The time and effort required to get the alert format right is substantial. Although it was certainly expected, the business return on the project was dependent to a large degree on the ability to deliver to the analysts information in a format that they could readily use. After quite a bit of experimentation, a report format was developed that reported:

- What is the issue? This part of the report identifies the relevant business unit and the relevant business issue.
- Who has the issue? This part of the report identifies the relevant subsystem of VisaNet, the relevant attribute, and the relevant attribute value.
- What is the business opportunity? This part of the report identifies the daily business value associated with the issue and the statistical significance of the alert (Low, Medium High).
- What is the business impact? This part of the report describes the a business measure as currently measured, the historical measure of the business measure during the baseline period, and the number of transactions affected.

The final part of the report contains additional information, such as the alert ID, alert creation date, whether the alert is new, and whether the alert is associated with an issue that has been previously identified and now is being monitored for compliance.

One way to summarize the report is that the items in alerts gradually changed from those items related to the statistical models and how the alerts were generated to items directly related to how the alerts were investigated and how the business impact was estimated. The surprise for us was not that this transition had to be made, but rather the time and effort required to get it right.

Lesson 3. It turned out that some of the most important alerts we found were alerts that had low statistical significance. For each report, we include an estimate of the statistical significance of the alert (low, medium, high and very high) as well as an estimate of the business significance of the alert (in dollars). It turned out that after investigation, the alerts that generated by most dollars saved, were often the alerts with low statistical significance. For this reason, it was usually not a good idea to investigate alerts in the order of most statistically significant to least statistically significant. Rather, the analysts used a more complex prioritization that thus far we have not tried to formalize.

Lesson 4. As a result of analysis of an alert, it was sometimes possible to create specialized baselines and reports that would look for similar problems in the future. We quickly learned that even a few specialized reports like this could easily occupy most of our available time. The lesson we learned was that it was important to devote some of our time to looking for new opportunities (think of this as a survey activity), since some of these turned out to be even more important than what we were currently doing.

10. RELATED WORK

There is a large amount of research on change detection algorithms per se. The monograph by Basseville and Nikiforov [3] is a good summary of this research field. In particular, the change detection algorithms that we use here, including CUSUMs, Generalized Likelihood Ratios, and related algorithms are covered in this reference.

The work described in this paper differs from classical change detection and contingency tables in that it uses a separate change detection model for each cell in a cube of models.

More recently, Ben-David, Gehrke and Kifer [4] introduced a non-parametric change detection algorithm that is designed for streams. The methods used here are parametric. In contrast to their approach which uses a single change detection model, we build a large number of models in order to handle complex, heterogeneous data, one for each cell in a multi-dimensional data cube.

The paper by Fawcett and Provost [7] has a similar goal – detecting unusual activity in large data sets – but uses a much different approach. Their approach is to introduce operating characteristic style measures in order to identify unusual behavior.

Guralnik and Srivastava [9] study event detection in time series data by using a new change detection algorithm they introduce, which involves iteratively deciding whether to split a time series interval to look for further changes.

In contrast to all these methods, our approach is to use relatively simple classical change detection algorithms, such as CUSUM and GLR, but to build thousands of them, one for each cell in a multi-dimensional data cube. As far as we are aware of, our paper is also one of the few papers in the data mining literature that presents a case study of change detection involving a system as large and heterogeneous as VisaNet.

11. SUMMARY AND CONCLUSION

In this paper, we have shared our experiences and some of the lessons learned over the past two years developing and operating a baseline and change detection system for Visa. Because of the complex and highly heterogeneous nature of Visa's transactional data, we did not build a single change detection model, but rather over 15,000 individual change detection models. Indeed we built a separate change detection model for each cell in a multi-dimensional data cube. This is an example of we have been calling Change Detection using Cubes of Models or CDCM.

Overall, the approach seems to work quite well. Indeed, substantial business value is being generated using this methodology, and thus far we have not been able to achieve the same performance using a single baseline or change detection model.

To summarize, we have demonstrated through this case study that change detection using data cubes of models (CDCM) is an effective framework for computing changes on large, complex data sets.

12. REFERENCES

[1] Alan Agresti, An Introduction to Categorical Data Analysis, John Wiley and Sons, Inc., New York, 1996.

[2] The Augustus open source data mining system can be downloaded from www.sourceforge.net/projects/augustus.

[3] M. Basseville and I. V. Nikiforov. Detection of Abrupt Changes: Theory and Application. Prentice Hall, 1993.

[4] Shai Ben-David, Johannes Gehrke, Daniel Kifer, Detecting Change in Data Streams, Proceedings of 2004 VLDB Conference, 2004.

[5] The Predictive Model Markup Language, Data Mining Group Version 3.1, retrieved from www.dmg.org on January 10, 2007.

[6] Joseph Bugajski, Robert Grossman, Eric Sumner, Tao Zhang, A Methodology for Establishing Information Quality Baselines for Complex, Distributed Systems, 10th International Conference on Information Quality (ICIQ), 2005.

[7] Tom Fawcett and Foster Provost, Activity monitoring: noticing interesting changes in behavior, KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, 53--62, ACM Press, New York, 1999.

[8] Robert L. Grossman, PMML Models for Detecting Changes, Proceedings of the KDD-05 Workshop on Data Mining Standards, Services and Platforms (DM-SSP 05), ACM Press, New York, 2005, pages 6-15.

[9] Valery Guralnik and Jaideep Srivastava, Event detection from time series data, Proceedings of the Fifth ACM SIGKDD international Conference on Knowledge Discovery and Data Mining, ACM Press, New York, NY, 33-42, 1999

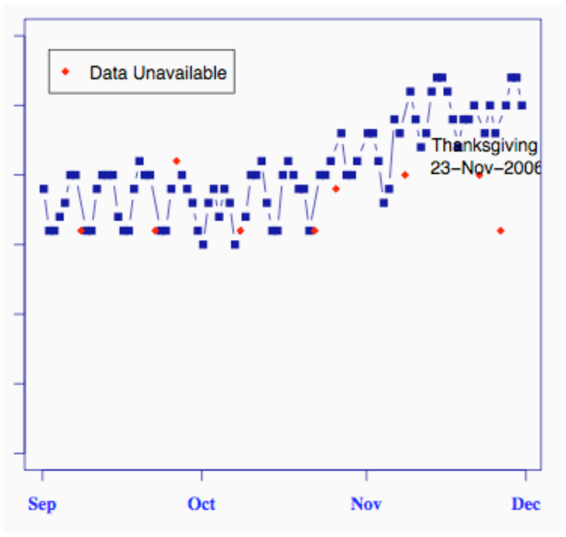

Figure 1. This is an example of one of the measures monitored in this project. This graph shows how the ratio of declined transactions varies for one of the Merchant Category Codes (MCC) monitored. Note the daily, weekly and monthly variation in the data. This variation, which is typical of the measures tracked, is one of the reasons detecting changes in this data is challenging. The vertical axis scale has been omitted due to confidentiality reasons.

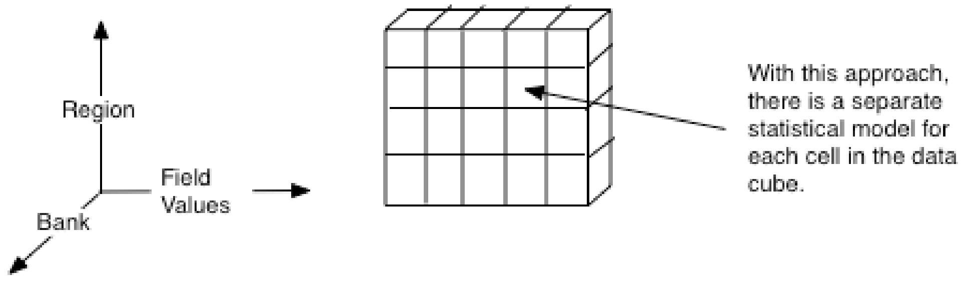
Region
Bank
Field
Values
With this approach, there is a separate statistical model for each cell in the data cube.

Figure 2. The basic idea with change detection using cubes of models orCDCM is that there is a separate change detection model for each cell in a multi-dimensional data cube. In the work described here we estimated and maintained over 15,000 different baseline statistical models and monitored them monthly.

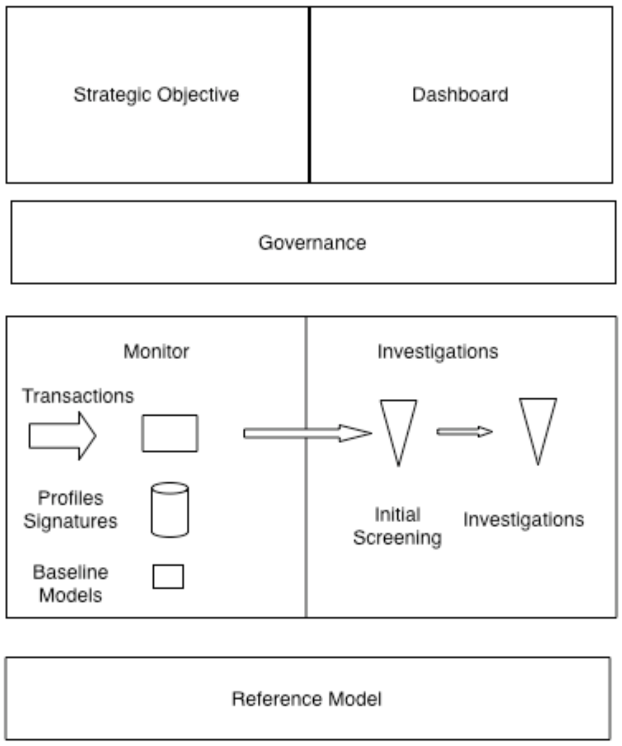

Figure 3. This figure summarizes some of the key components of the program and process set up to detect data quality and data interoperability problems. Note that the baseline model and monitor are just two of the components.

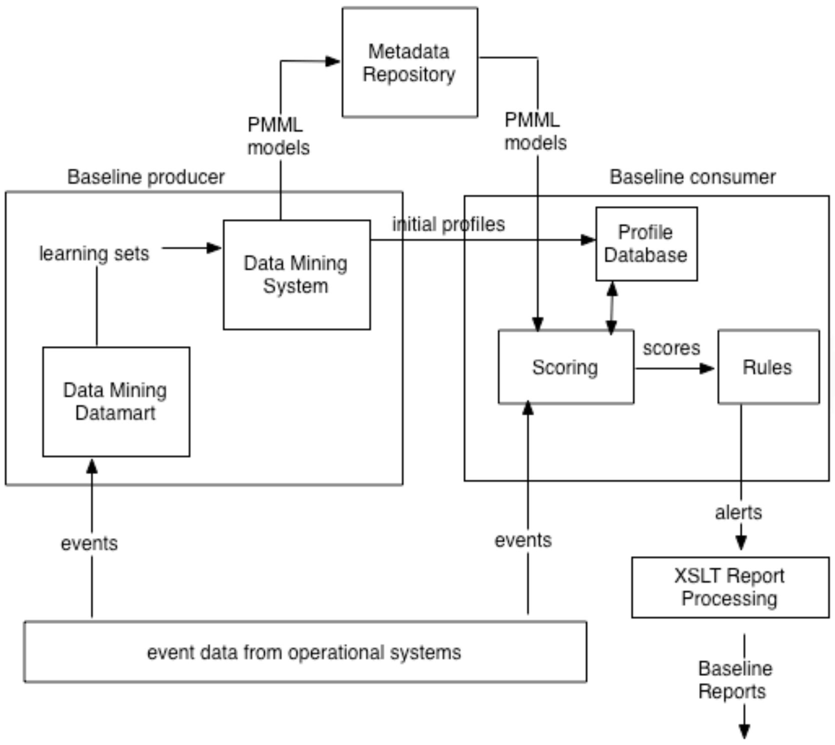

Figure 4. This figure shows the architecture we used to compute the baseline alerts