Chapter 14

Fraud Detection

Corruption, embezzlement, fraud, these are all characteristics which exist everywhere. It is regrettably the way human nature functions, whether we like it or not. What successful economies do is keep it to a minimum. No one has ever eliminated any of that stuff.

– Alan Greenspan

Contents

On July 18, 2005, the New York Times ran a story with the headline "New York Medicaid Fraud May Reach Into Billions"[80]. The authors, Clifford J. Levy and Michael Luo, reported that "New York State's Medicaid program had become a $44.5 billion target for the unscrupulous and the opportunistic."

Medicaid is a federal program that was adopted in New York State in the 1960s to provide medical coverage for residents who were not covered by any other medical insurance program and whose income was below a prescribed threshold. The annual cost of this program for New York State had grown to $44.5 billion by 2005. Private medical insurance companies had already found that consistently 10% to 20% of their claims should not have been paid because they were either fraudulent or because the procedures were of no value. (The latter situation often occurred when prescribed tests and procedures were rendered obsolete by new medical developments.) Based on this, it was recognized that the annual loss to New York State due to fraud and abuse was many billions of dollars.

Levy and Luo used the state Freedom of Information Law to access information on payments made to physicians by Medicaid. They discovered extraordinarily large payments made to some providers; over one million dollars of these were deemed to be fraudulent. These included a dentist operating out of a Brooklyn storefront who claimed to have performed as many as 991 procedures one day in 2003. Her dental practice had grown to become the state's biggest Medicaid dental practice within one year. The New York Times investigation also discovered abuses in speech therapy, nursing home operations and in the diverting of drugs intended for AIDS patients to bodybuilders.

On July 19, 2005, Governor George Pataki of New York ordered a broad overhaul of state agencies that protect Medicaid from fraud and abuse, creating an independent inspector general's office and bringing in a former federal prosecutor to help reorganize the policing of the problem.

In this chapter, we discuss how a computer based analytics system can work with a trained human auditor to identify claims that should not be paid. We also describe experiences encountered when IBM's Center for Business Optimization used this system working with New York State to combat Medicaid fraud.

14.1 Reducing Health Care Fraud

There are many issues encountered when trying to reduce losses due to fraud and abuse in health care. A first challenge is detection: discovering which claims are likely to be fraudulent and which providers should be investigated. In some cases, such as a dentist claiming to have performed 991 procedures in a single day, this is not difficult. However, in other cases which are not as blatant, it becomes much more difficult to identify which claims and claimants are suspicious. Fraudsters often will try to hide spurious claims by inserting them with large groups of valid claims. It is often only by examining large

numbers of transactions in the context of other claims that investigators can detect those that seem suspicious.

A second challenge is deciding which cases should be pursued and which of these cases should be taken to court. Investigations themselves are expensive and take time. The overall goal is to reduce unnecessary costs incurred by the program. If the savings resulting from an investigation are less than the cost of the investigation, then the process may be deemed counterproductive.

A third challenge is recovering payments which should not have been made. The traditional approach has been termed "chase and pay." This consists of detecting an improper payment after the fact and then attempting to recover the money. However, in many cases, it can be difficult or impossible to recover such payments. A more appealing solution is to detect fraudulent claims before payments are made and deny them. However, this has to be reconciled with the requirement that legitimate health care provider claims should be paid promptly.

A fourth challenge is that the world is constantly changing. Any system developed to detect fraudulent claims will need to adapt as quickly as fraudsters can invent new ways to cheat the system.

In general, there are two main ways of abusing a program. One is by unauthorized persons masquerading as legitimate claimants to a system and submitting claims that would be legitimate if they were submitted by a valid claimant. For example, a person who is not a medical doctor (MD) should not be able to be paid for claims that can only come from MDs.

The second occurs when a legitimate claimant makes a claim that should be rejected for any of several reasons. This can be a complex issue for Medicaid fraud prevention, because there are so many types of legitimate claims. As an example of this type of fraud, Levy and Luo reported that ambulettes (specially equipped vans for transporting patients unable to walk unaided in nonemergency circumstances) cost taxpayers up to $200 million per year. They regularly submitted claims for transporting scores of people who walked quite easily while being observed by a reporter. Moreover, in some cases, the rides that the state paid for may have never taken place.

The first problem, identifying a claimant and determining whether they are entitled to make a claim, is covered by a range of biometrics and knowledge based identifiers and we do not address it in this chapter. Our focus is on the second form of abuse: How can we tell whether a claim submitted by a legitimate claimant is itself valid and should be paid? Can we identify which legitimate claimants are submitting invalid claims?

14.2 Detection and Prevention of Medical Fraud and Abuse

Three main parties are involved in the providing of health care to insured patients: These are (a) *service providers*, which includes doctors, hospitals,

ambulance companies, and laboratories; (b) *insurance subscribers*, which includes patients and patients' employers; and (c) *insurance carriers*, who receive regular premiums from their subscribers and pay health care costs on behalf of their subscribers. This last group may include governmental health departments as well as private insurance companies.

Li et al [83] report that the largest source of fraud is due to service providers. This is primarily due to billing for items, such as services not delivered or for procedures such as lab tests not actually obtained. This may place patients at risk and is recognized as the most urgent problem for improving the quality and safety of a health care system.

An insurance carrier receives tens of thousands of claims daily from providers. Most of these are legitimate but some are not, and should be rejected. If a claim is legitimate, it is important that it is paid promptly. If not, service providers may decide to stop accepting that company's coverage. If it has been paid, and subsequent analysis indicated that this was a mistake, then it may or may not be possible to recover the funds, but in either case, this may be difficult and very costly.

It is not expected that an analytic system will be able to correctly classify all claims. The goal of these systems is to quickly identify claims that are sufficiently suspicious that they warrant investigation by a human expert. An ideal system will keep the number of *false positives* – valid claims flagged for investigation – and *false negatives* – invalid claims approved for payment – below acceptable thresholds. Every false positive incurs an investigative cost, with no resulting recovery, and has a negative impact on the claimant's satisfaction with the company. Every false negative results in an invalid payment being made and which may be difficult or impossible to recover. It is easy to create a system that guarantees zero false positives by simply rejecting all claims, or zero false negatives by simply paying all claims. The difficulty is designing a system which maintains an acceptable balance between the two.

Determining an acceptable balance between false positives and false negatives depends mainly on the consequences of each type of error. Recovering from a false negative may require collecting invalid payments that were made. A false positive may be difficult to recover from if it results in service providers leaving the network.

In addition, claims selected for investigation should be prioritized based on the expected amount of funds recoverable. This depends on both the potential recovery from a paid fraudulent claim and the probability of a successful recovery of the funds.

The financial importance of reducing fraud and abuse can be significant. A major insurance carrier reported a net income from continuing operations of \$1.3 billion in 2011. The total adjusted claims they paid was \$0.75 billion. If 20% of these claims should not have been paid, this would have moved \$0.15 billion from the cost line to the net income line in the annual report, an increase of approximately 3%.

A successful fraud prevention and recovery program should have the following characteristics:

- Claims are analyzed quickly enough to avoid unacceptable delays in payment of valid claims.
- The numbers of false positives and false negatives are kept below acceptable limits.
- Claims identified as suspect and passed on to investigators have sufficient value to justify the cost of investigation.
- Investigators are provided with data-based justification for rejecting the suspected fraudulent claims.
- The system is adaptable and able to respond quickly to new types of fraudulent claims.

The simplest types of fraud detection systems are *rule based systems.* In this case human domain experts create sets of rules that they expect valid claims to satisfy. For example, a dentist should not see more than twenty-five patients per day. A person living in a certain town should not live in a multi-million dollar home. A patient receiving treatment for hypertension should not require x-rays. A patient being treated for high cholesterol should not see a doctor more than twice per month.

Violating these rules may or may not indicate fraud or abuse. In some cases there may be a valid reason. For example, the person receiving treatment for hypertension may, at the same time, be diagnosed for another condition.

Claims are then processed by verifying that all or most of the rules defining acceptability are satisfied. If they are, the claim is paid. If not, the claim is rejected.

There are several difficulties with such a system. First, it is difficult to define effective sets of rules, especially for a rapidly evolving field such as medicine, where new processes, treatments and norms require the rule sets to be continually revised and updated. Second, the rules generally try to encapsulate the intuition of human experts. Often fraud is carried out in ways that avoid extreme exceptions to normal behavior. For example, a small number of fraudulent claims per month may pass through undetected. Third, it is difficult to detect new approaches to fraud.

Instead, data based fraud detection systems operate by assembling large collections of provider, insurer and patient data. They take as input records giving detailed information on all claims, healthcare providers and other medical factors. Domain experts then work interactively with the system to identify clusters of records, called *peer groups*, that should have similar characteristics. For example, cardiologists working in clinics of comparable sizes serving patients in the same communities could form a peer group.

After peer groups have been identified, a range of statistical measurements can be applied to the data in each group to determine normal behavior

and identify statistical outliers. These become potential candidates for investigation.

Finding peer groups is an example of *unsupervised learning.* Unsupervised learning refers to approaches which do not depend on experts carrying out a prior classification of all claims in the training set as either valid or fraudulent before acceptance criteria are developed. Rather, they propose peer groups in the unclassified data and look for statistical outliers.

Targets for investigation are differentiated based on whether they are identified before or after payment of the claim has been made. *Forensic analysis* applies to claims which have already been paid. *Preventative analysis* reviews claims submitted but not yet paid. For the latter class of claims, speed of analysis is critical. For the former, not only must we take into account the probability of a claim in the cluster being fraudulent or improper for reimbursement, but also the size of the claim and the probability of a successful recovery.

14.3 Defining Peer Groups

The identification of peer groups uses *clustering*, an important class of analytics methods falling under the category of *unsupervised learning.* The input is normally a large database of records and the objective is to partition the records into smaller sets called *clusters* or *peer groups* all of which are "similar" and such that records in different clusters are "dissimilar." There are many different ways of clustering records, depending primarily on the choice and relative importance of attributes considered when determining the difference between two records. We provide general information on clustering in Chapter 21.

Records related to medical claims could contain the following data elements:

- Provider ID
- Provider Specialty Code
- Provider Address
- ZIP code of Provider
- Patient ID
- Patient ZIP Code
- Code for Service Provided
- Usual time claimed for service provided
- Date and time of service
- Charge for service

- Lab tests prescribed
- Pharmaceuticals prescribed

Suppose that we had data available for all medical claims received by an insurance company over a year. We could cluster the data in many different ways, based on (a) the provider ID or the Patient ID, (b) the provider address or the patient address, (c) the provider specialty, or (d) the month when the service was provided.

However, more useful clusters are often obtained by combining criteria. Suppose we clustered based on the combination of Provider Specialty Code, Code of Service Provided and Patient ZIP Code. This would enable us to consider data corresponding to patients living in different geographic areas who had certain services provided by different types of specialists. After clustering, we could try to identify "normal" behavior. For example, we might discover that in ZIP code 10589, patients seeing a certain specialty provider have certain types of services provided more frequently than those in a neighboring ZIP code.

When clustering criteria are chosen, we are then able to characterize common and unusual characteristics within each cluster. In the above example, if the number of bills with procedure code BX23 was, on average, three per month and provider 2642 billed for twenty-four per month, this would be detected as unusual behavior. This does not necessarily mean that there is fraud, but it does indicate that this could be a case worth investigating.

A key to success is choosing suitable clusters. If the clusters are too large, then it may be difficult to identify outliers for investigation. If, in the above example, we had grouped all specialties of providers together, then we might have found that within this larger group, twenty-four billings of procedure BX23 is normal, but when restricted based on the providers' specialty, it stands out.

There are many different ways to define clusters on a set of data. Usually, a first step is to decide which attributes are likely to be relevant for the subsequent analysis to be performed. For example, the number of doctor visits per year by a particular patient could be important but the patient's middle initial could be deemed to be irrelevant. Clustering is an iterative process and frequently, based on results obtained, the selected attributes will be changed and clustering will be redone.

It may also be advantageous to define new attributes which are composites of attributes coming in the raw data. Domain experts may know that a certain type of lab test is normally applied to specific types of condition and so may define a new attribute combing the test and the condition.

We define a *distance* or *dissimilarity index* between each pair of data records, based on the attributes that we wish to consider. In the above example, the distance between records s and t could be defined as follows:

Let $Specialty(s,t) = 0$ if s and t have identical Provider Specialty Codes and 10 if they are different.

Let $Service(s,t) = 0$ if s and t have identical Codes of Services Provided, 1 if the services are different but similar and 5 if they are completely different.

Let $ZIP(s,t) = 0$ if the ZIP codes are identical, 5 if they are different but in the same state and 10 if they are in different states.

The distance between records s and t could then be defined as

$$d(s,t) = Specialty(s,t) + Service(s,t) + ZIP(s,t).$$

More generally, we would define $d(s,t)$ to be a weighted sum of the attribute distance values, where the weights are determined by combining the assessments of domain experts and data analysts.

The goals are (1) for the members of each cluster to be sufficiently similar in the selected attributes that we could expect them to have similar billing characteristics, and (2) for the members of different clusters to be sufficiently different in their billing characteristics. These goals are often contradictory.

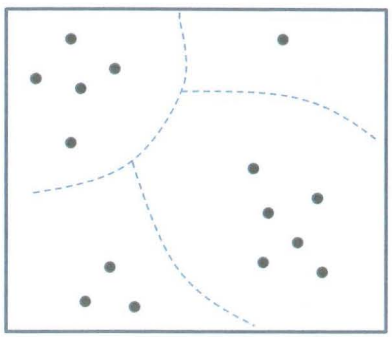
There are many different approaches to clustering; different methods may work better on different data sets. In practice a data analyst will often work together with a domain expert to find a combination of distance measure and clustering method which works well for the particular application.

Figures 14.1 and 14.2 illustrate some of the issues encountered with clustering. We have represented fifteen records by points where the measure $d(s,t)$ of the distance between records s and t is proportional to the Euclidean distance between the points representing the records.

Here are two trivial clusterings. One consists of a single cluster containing all the points. The other consists of fifteen clusters, each containing a single point. A possibly more useful clustering, shown in Figure 14.1, divides the points into four clusters containing five, one, six and three points.

In practice, we often want a clustering to satisfy additional properties. These could include upper and/or lower bounds on the number of points in each cluster or the total number of clusters.

Figure 14.1: Set of points representing 15 records and a "natural" clustering.



In Figure 14.2 we show a clustering satisfying the constraint that each set should contain at least four points. This constraint results in a clustering that looks less natural.

When we are using clustering to construct peer groups for fraud analysis, we generally do not have rigid size criteria. However, we do want the groups to be of a sufficient size to permit valid statistical analysis but still small enough to permit detection of a reasonable number of outliers across the entire population.

After a distance $d(s,t)$ between individual records s and t has been defined, we can extend the definition of d to a distance D between sets S and T of records in several ways. For example, the distance between two sets could be defined as the distance between the nearest pair of points in the two sets, the distance between the most distant pair of points in the two sets or the average distance between all pairs of points in the two sets.

The *diameter* of a cluster S is the maximum value of $d(s,t)$ for all $s,t \in S$.

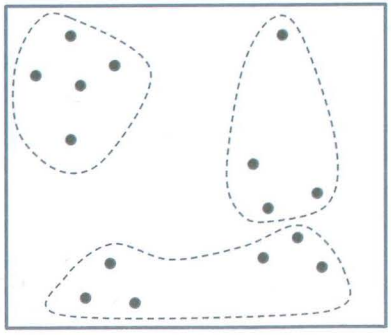
The overall goal in clustering is to find a collection of clusters for which

1. the diameters are small;
2. the distances between clusters is large;
3. the sets satisfy any required size constraints;
4. the number of sets in the cluster is acceptable.

There are several general approaches. One is a "bottom-up" approach called *agglomerative hierarchical clustering.* Initially each point is a one element cluster. Then two existing clusters S and T, for which the selected distance measure is minimum, are repeatedly combined to create a larger cluster.

Another approach, called *divisive hierarchical clustering*, is "top-down." Start with a single cluster that contains all the points. Repeat the following operation until done: For each existing cluster which is too large, find a way to split the cluster into two smaller clusters S and T so that you maximize

Figure 14.2: Clustering with all sets containing at least 4 points.



the distance between them. See Chapter 21 for further discussion of these approaches.

The use of clustering in fraud detection applications is to partition the data records into clusters for which the records have a high degree of similarity for a selected set of attributes. We can then apply statistical analysis to each cluster to understand typical behavior. In the Specialty/Service/Zip example, we could determine the average amounts of billings. We could also determine the average number of patients seen per day or the most common prescriptions associated with these services. Or, we could discover the extent to which these values are correlated with where a patient lives.

For fraud detection, the goal of clustering is to create clusters for which statistical outliers in the clusters are suitable targets for investigation. For example, these outliers could be healthcare providers seeing an unusually large number of patients per day or those who bill for an unusually large number of services, compared to other members of their peer group. In the next section, we describe the method of clustering that was the most effective in practice in creating peer groups which enabled identification of "interesting" outliers.

14.4 The Condorcet Method

In this section, we describe a clustering technique developed by Pierre Michaud and Jean-François Marcotorchino of IBM France [89]. It has been referred to by various names: *The Condorcet Method*, *Similarity Aggregation*, or *Relational Clustering*. It has been found to be very effective for performing clustering which enables detection of suspicious claims.

Suppose we are given a set R of n records. Each record contains the same sequence of m attributes (fields). An analyst typically begins by selecting a subset A of the attributes as the basis for the clustering. Let v be the number of attributes selected.

If the data is presented as a spreadsheet, then the records correspond to the rows, the attributes correspond to the columns and A is the set of v columns selected as the basis for the clustering. Let A_j be the set of all possible values for the j^{th} attribute that can occur in the records.

Recall that a clustering is a partition of all records R into a number of pairwise disjoint sets called clusters. For *categorical* attributes the set of possible values in A_j is discrete and hopefully of a manageable size for each j. These contrast with *continuous* attributes which are allowed to take any value within a prescribed range. If A_j is categorical with a large number of categories, or continuous, we can perform a preliminary process called *bucketing*. We partition A_j into some manageable number of subsets and replace the original attribute with the index of the subset to which it belongs.

We can define a trivial clustering based on any single attribute. For each record r, we let r_j denote the value of its j^{th} attribute. We put records r and s in the same cluster if $r_j = s_j$ and in different clusters if $r_j \neq s_j$. This defines

a clustering consisting of at most $|A_j|$ clusters based solely on the value of the j^{th} attribute but which completely ignores all other attributes. It is generally not of much use.

The Condorcet Method has two goals: The first is to define a distance measure that provides a "reasonable" combination of these trivial clusterings. The second is to develop an effective method of constructing a "good" clustering.

Distance and Optimal Clusterings

For each pair (r, s) of records, we let $d(r, s)$ be the number of attributes for which r and s *disagree*. We then have $0 \leq d(r, s) \leq v$ for every pair (r, s) of records, $d(r, s) = 0$ if r and s agree for all attributes in A and $d(r, s) = v$ if they disagree for all attributes in A.

We specify a target L satisfying $0 \leq L \leq v$. The goal is to create clusters satisfying the following conditions:

$$\text{If } d(r, s) \leq L, \text{ then } r \text{ and } s \text{ are in the same cluster;} \tag{14.1}$$

$$\text{If } d(r, s) > L, \text{ then } r \text{ and } s \text{ are in different clusters.} \tag{14.2}$$

In general, it is not possible to simultaneously satisfy these conditions for all pairs of records. Our goal is to efficiently find a clustering for which the amount of violation of (14.1) and (14.2) is small, ideally as close to the minimum as possible. This we do by computing a penalty $p(r, s)$ for each pair of records which violate (14.1) and (14.2) equal to $d(r, s) - L$ if r and s are in the same cluster and $d(r, s) > L$, or equal to $L - d(r, s)$ if $d(r, s) \leq L$ and r and s are in different clusters. We then try to construct a clustering which minimizes the sums of these penalties over all pairs $\{r, s\}$ of records.

For example, in Figure 14.3, the nodes represent four records r_1, r_2, r_3 and r_4. The values on the edge joining each pair of nodes $\{r_i, r_j\}$ represent the disagreement $d(r_i, r_j)$ between r_i and r_j. Suppose $L = 5$. The low disagreement on the edges r_1r_2, r_2r_3 and r_3r_4 indicates that we should have a single cluster consisting of all the records. The high disagreement on the edges r_1r_3, r_2r_4 and r_1r_4 indicates that we do not want a single cluster containing any of these pairs of records. Any way we cluster these records, we will either have two records with disagreement less than 5 in different clusters or two nodes with disagreement greater than 5 in the same cluster.

We blend the two conditions (14.1) and (14.2) by defining an aggregate measure:

$$D(\mathcal{C}) = \sum (d(r, s) - L : r, s \text{ are in the same cluster of } \mathcal{C} \text{ and } d(r, s) > L)$$

$$+ \sum (L - d(r, s) : r, s \text{ are in different clusters of } \mathcal{C} \text{ and } d(r, s) \leq L). \tag{14.3}$$

Note how (14.3) takes into account both disagreement between pairs of records in the same cluster and agreement between pairs of records in different clusters.

Our objective then is to find a clustering $\mathcal{C}$ for which $D(\mathcal{C})$ is as small as possible.

In the example of Figure 14.3, if $L = 5$, then the optimum clustering is $\mathcal{C} = \{\{r_1, r_2\}, \{r_3, r_4\}\}$ and $D(\mathcal{C}) = 2$. If $L = 8$, then the cost of this clustering would become $8 = (8 - 3) + (8 - 6) + (8 - 7)$. An optimal clustering $\mathcal{C}$ for this case would be a single cluster containing all the records and $D(\mathcal{C}) = 2$.

The second requirement for Condorcet clustering is an effective method of constructing a clustering $\mathcal{C}$ which minimizes, or comes close to minimizing, $D(\mathcal{C})$. Michaud and Marcotorchino proposed two approaches. The first is an exact optimization method which computes a clustering which minimizes $D(\mathcal{C})$. The second method is a heuristic which works well in practice but cannot be guaranteed to find a cluster $\mathcal{C}$ which actually minimizes $D(\mathcal{C})$.

The first method has the disadvantage that the time required can increase significantly as the number of records grows. This presents a problem because typically we have very large sets of records for fraud analysis because there is one for each transaction. We outline this method at the end of this section.

We focus here on the second, heuristic, method. It works well in practice, is relatively fast and is the method used in the software systems we describe in Section 14.6.

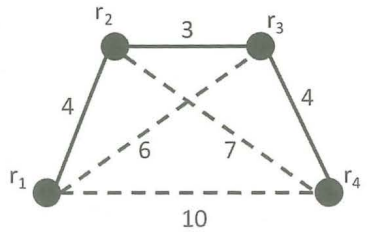
Condorcet Clustering Heuristic

We defined the distance $d(r, s)$ between records r and s as the number of attributes we are using for clustering for which the records have different values. We now define the distance $d(r, K)$ of record r from a cluster K as

$$d(r, K) = \sum (d(r, s) - L : s \in K \text{ and } d(r, s) > L)$$

$$+ \sum (L - d(r, s) : s \notin K \text{ and } d(r, s) \leq L). \tag{14.4}$$

Figure 14.3: Conflicting values of $d(r_i, r_j)$.



This captures the fact that if we add r to K, we incur a cost of $d(r,s) - L$ for each $s \in K$ such that $d(r,s) > L$, and incur a cost of $L - d(r,s)$ for each s such that $s \notin K$ and $d(r,s) \leq L$.

The Condorcet clustering heuristic has two phases. In the first phase it constructs a starting clustering $\mathcal{C}$ in a greedy fashion. In the second phase it repeatedly performs an improvement process wherein it makes local improvements to the current clustering $\mathcal{C}$ which reduce $D(\mathcal{C})$.

Initialize: Let $\mathcal{C} = \emptyset$ be the empty clustering.

Process each record $r \in R$ in turn.

If there exists a cluster K in $\mathcal{C}$ for which $d(r,K) \leq L$, then choose such a K for which $d(r,K)$ is minimum. Add r to K.

If there is no cluster $K \in \mathcal{C}$ for which $d(r,K) \leq L$, then we form a new cluster $K' = \{r\}$ and let $\mathcal{C} := \mathcal{C} \cup \{K'\}$.

Local Improvement: Process each record r in turn.

See whether $D(\mathcal{C})$ would be reduced by removing r from the set K containing it and adding it to a different set $K' \in \mathcal{C}$ or letting r be a new (singleton) cluster in the clustering.

If so, choose the best such option. If not, do nothing.

Repeat the Local Improvement step: Until no sufficiently large reduction in $D'(\mathcal{C})$ is obtained.

In practice, usually after a small number of repetitions (four or five) of the Local Improvement cycle, the reduction in $D(\mathcal{C})$ at each subsequent step becomes very small. At this point, we consider the clustering step to be complete and terminate this part of the process.

There are several possible variations to this method. Instead of simply defining d_{st} to be the number of attributes for which r_s and r_t differ, we could let d_{st} be the sum of measures of the differences between the attribute values using some suitable scale. We could also make d_{st} a weighted sum of the measures of these differences.

We could modify the Condorcet clustering algorithm local improvement step by "parking" any records that the algorithm selects to start a new singleton cluster. Then perform a second pass on these parked records after all other records have been processed. This can have the effect of delaying difficult decisions until many straightforward decisions have been made.

However, there are a couple of caveats: First, different variations may work better on some data and worse on other data. The best way to evaluate variations is to try them on real datasets and compare the outcomes. Second, the purpose of the clustering is to enable us to find outliers as described in the next section. Sometimes a clustering can be worse when evaluated by the clustering criterion, but still may enable us to find more interesting outliers.

Optimal Clustering*

The Condorcet clustering problem can be solved as a mixed integer optimization problem [89]. Recall that we are given a set R of n records and for each pair $\{r, s\}$ of records we have defined a distance $d(r, s)$ between the two records based on our selection of attributes, relative weightings of the attributes and the costs we assign to a difference in the values of each attributes for the two records.

We want to create a clustering of a set of the records in R based on $d(r, s)$. We can represent a clustering by a $n \times n$ (0,1)-matrix $X = (x_{rs} : r, s \in R)$ with the interpretation that $x_{rs} = 1$, if r and s are in the same cluster, and $x_{rs} = 0$, otherwise. We impose the following constraints on X:

- Reflexivity: (Each point is in the same cluster as itself.)

$$x_{ss} = 1 \text{ for all } s \in R, \tag{14.5}$$

- Symmetry: (If records r and s are in the same cluster, then both x_{rs} and x_{sr} have value 1.)

$$x_{rs} = x_{sr} \text{ for all } s, t \in R, \tag{14.6}$$

- Transitivity: (If r and s are in the same cluster and s and t are in the same cluster, then r and t are in the same cluster.)

$$x_{rs} + x_{st} - x_{rt} \leq 1 \text{ for all } r, s, t. \tag{14.7}$$

Any (0-1)-matrix $X = (x_{rs})$ satisfying (14.5), (14.6), and (14.7) defines the classes of an equivalence relation $\mathcal{R}$ defined on the records where $r\mathcal{R}s$ if and only if $x_{rs} = 1$. The clusters (peer groups) are the equivalence classes of $\mathcal{R}$.

Recall that we are also given a target value L. Our overall objective is for any pair $\{r, s\}$ of records to be in the same cluster if $d(r, s) < L$ and in different clusters if $d(r, s) \geq L$. To this end, for each $\{r, s\}$, we define a penalty p_{rs} to be incurred by any pair of records which violate this condition as follows:

$$p_{rs} = \begin{cases} d(r,s) - L, & \text{if } x_{rs} = 1 \text{ and } d(r,s) > L, \\ & [r \text{ and } s \text{ are in the same cluster}], \\ L - d(r,s), & \text{if } x_{rs} = 0 \text{ and } d(r,s) \leq L, \\ & [r \text{ and } s \text{ are in different clusters}]. \end{cases} \tag{14.8}$$

Our optimization problem then becomes:

$$\begin{array}{ll} \text{minimize} & \sum(p_{rs} : r, s \in R) \\ \text{subject to} & (14.5),\ (14.6) \text{ and } (14.7) \\ & p_{rs} \geq (d(r,s) - L) - M(1 - x_{rs}), \quad \text{for all } r, s \in R \\ & p_{rs} \geq (L - d(r,s)) - M(x_{rs}), \quad \text{for all } r, s \in R \\ & x_{rs} \in \{0, 1\} \quad \text{for all } r, s \in R \\ & p_{rs} \geq 0 \quad \text{for all } r, s \in R, \end{array} \tag{14.9}$$

where M is a value at least as large as the maximum value of $d(r, s)$ over all pairs r, s of records.

As mentioned earlier, this approach could be used if the number of records is small, but will become computationally intractable if the number is large.

14.5 Detecting Suspicious Behavior

Assume that peer groups (clusters) have been created based on some combination of selected attributes. Now each cluster is analyzed to determine "normal" behavior and, more importantly, outlier behavior. For example, we might decide to investigate the total annual billings for all providers belonging to a selected peer group. We compute the mean μ and the standard deviation σ of these total billings. If the set of total billings for all providers belonging to the group is normally distributed, then about 99.7% of the values would lie within three standard deviations of the mean. Suppose that $\mu = \$180{,}000$ and $\sigma = \$20{,}000$. If we wanted to investigate those providers whose billings are in the top 0.15% of the population, we would investigate those providers billing more than $240,000.

This might produce too few or too many targets for investigation. In this case we could increase or decrease the limit of $240,000 as necessary to get a suitable number. By determining how many standard deviations from the mean the new target lies, we can estimate the fraction of the population that is targeted.

Determination of outlier behavior for a peer group begins by defining a target value which is computed from an attribute or a combination of attributes. This is usually done by a domain expert who is familiar with the problem being solved and can apply judgment as to what could indicate fraudulent behavior. For example, a target value could be the ratio of the annual tax paid by providers to the total amount of their annual billings. In this case, a small value might warrant investigation. The actual selection of target value is often as much an art as a science and depends on the skill and experience of the domain expert.

There still remains an important question: What is an appropriate model for the distribution of the target value? For example, a certain target value could be modeled by a bell curve, a skewed distribution or a fat tail distribution. There are statistical measures for evaluating how well a model of a certain type matches the data but often the most useful method is a dashboard. This provides a way for an analyst to view a number of different visualizations of the data on a screen at the same time. These visualizations can include a display of different distributions applied to sets of different possible target values. The analyst(s) can then select suitable models and target values. Then, based on these, outliers can be selected for investigation. Sometimes certain candidates for investigation show up with several different data models and target values. This helps give more confidence in the selection of providers.

This approach for selection of investigation targets has several important features. First, it can be repeated as often as desired, which means that if characteristics of records change, for example, claimants change some characteristics of their claims, then the peer groups can be re-created and possibly different outliers will be selected. Second, it becomes difficult for fraudsters to attempt to game the system because they do not know the basis for formation of the peer groups or of the selection of outliers belonging to the peer groups.

14.6 A Commercial Software Offering

Many fraud detection systems originated in the private sector and were subsequently adapted to the public sector. One such system is IBM's Fraud and Abuse Management System (FAMS). It was created in the early 1990s in partnership with CIGNA, a major health insurance provider, and has been refined continually since then. After it was deployed by several health insurance companies it was recognized that the analytic engine that had been created, called the Entity Profile Management System (EPMS), could be used for other applications in the area of fraud detection. These applications include detection of under reporting of tax (tax fraud) and detection of insider fraud within a company. This last application deals with a particularly problematic situation: trusted employees committing crimes misusing customer information. These are outlined in Section 14.8.

The basis of EPMS is peer group profiling. This is the process of describing sets of individuals, providers, insurers and transactions by sets of features and scoring these features individually based on their distributions.

The system supports a variety of distribution curves. These include bell curves, inverted bell distributions and a variety of distributions exhibiting left and right skews. For certain features, high values may represent an outlier. An example is the number of insurance claims by a provider. For other features, ultra low values may be suspicious. For example, an unusually small number of prescribed lab tests for patients having large amounts of claims for a certain medical diagnosis could be unusual. In the case of bell curves, either high or low values would be considered suspicious.

EPMS is designed to be used by domain experts who are familiar with the area of medical fraud, or more generally, with whatever type of fraud is being prevented. It provides a dashboard style interface which enables the expert to explore different combinations of attributes, score them based on the selected distribution curves, and thereby identify peer groups that are highly likely to exhibit fraudulent behavior.

The features are placed into business groups, for example, based on demographics, insurance claims, or historical records. These groups are not used in the evaluation of a combination of attributes, but are more amenable for human consumption. Each group is scored based on the associated distribution

curves and the group score may be weighted based on the expert's experience and domain knowledge. These groups are then summarized into a composite score for a set of individuals, providers, insurers or transactions.

Rank based scoring provides a new algorithm used to generate feature, group and composite scores. The score is associated with the rank, the ordinal position, of a value in a sorted list. Curves such as Growth, Decline, Bell and U-Bell are still applied to each feature. In the case of a growth curve applied to a feature, as values reach the most extreme value they will receive higher scores. EPMS uses two algorithms. One calculates the score of a feature. The other algorithm combines these scores and weights into a group or composite score.

A high score could be triggered by a single extreme attribute and would result in a high score. However, a combination of non-extreme attributes could also produce a high score. This is important. Criminals who commit a small number of big crimes are easy to detect. What is more difficult to detect are people whose records contain a small number of slightly abnormal features over a long period of time. An example of this is robbing a bank! Skimming a fraction of a cent off every transaction is not easily detected, but can result in significant gain over time. Persons committing a single large robbery usually end up in jail, but criminals who skim a small amount of money over a long period of time are much harder to detect.

The scores are then scaled to integers in the range 0 to 1,000, where 1,000 denotes the most extreme outlier. They are presented from high to low because the developers found that users seemed to respond better to that range than the range of continuous values between 0 and 1 normally used by statisticians.

The purpose of the score is not to make a judgment but to provide a prioritized list to the investigating officer (auditor) who is the final decider in the overall process. This investigating officer is the most expensive and difficult to obtain resource. The business value of EPMS is to make optimal use of this limited resource.

Companies view reduction of waste due to fraud and abuse as a common goal and, for that reason, they are willing to share their experiences and findings with other companies. For this reason, a FAMS users group was created which permitted the exchange of usage tips as well as the effective peer groups that were discovered and what was deemed normal behavior. This was important because often claimants who were abusing the system would apply the same approaches to a variety of insurance carriers. Sharing the details of detected fraud and abuse could significantly reduce the time required to detect and combat new types of fraudulent claims.

14.7 Verify New York

In 2005, the State of New York wanted to find ways to reduce the $44.5 billion of annual medicaid expenditures without sacrificing the quality of the program.

Reducing fraud and waste was an attractive target. However, there were certain issues that needed to be addressed if the system was going to work in that environment.

A challenge was that in New York State, the medicaid program is handled independently by each of the sixty-two counties, with some coordination at the state level. The level of sophistication of the counties with advanced analytics solutions was highly variable. As a result, the decision was made to pilot a new version of FAMS which would have three features: First, it would be tuned to handle Medicaid claims. Second, it would access data from the state medical repository when requested and authorized by a county. Third, the results of the analysis which would be applied on a per county basis would be performed by a secure hosted service provided by IBM. The results of the analysis would be delivered to the counties in the form of written reports. This system was called "Verify NY."

On September 15th, 2005 State Senator Thomas Morahan posted an announcement that Rockland County was piloting IBM's fraud detection software to detect Medicaid fraud and abuse. The month after this announcement, the value of medicaid claims in Rockland County decreased by approximately 15%, even though no fraud investigations had been completed. A news story announcing the introduction of an anti-fraud system can itself have a significant deterrent effect.

The delivery of the system as a hosted service made it much more cost effective than it would have been to install and maintain separate systems in each of the sixty-two counties. However, it also gave the developers the new challenge of how to present the results in a form that could be used by human investigators. Not only did this system need to identify outliers but it also had to explain, in human understandable terms, what made that provider an outlier.

14.8 Detecting Tax Fraud and Insider Fraud

Following the creation of EPMS/FAMS and its successful use in combating health care fraud and abuse, other applications were developed which used EPMS to detect unusual behavior in other areas. It was used to identify income tax filers who seemed to be underreporting income and tax paid. The approach was similar to what had been done for detecting health care fraud. Tax filers were clustered into peer groups based on a variety of attributes. Standard behavior was determined for each group and outliers were identified.

A different application was to combat insider fraud within a company. For example, a bank teller may look up Social Security Numbers for several clients in the course of a day. However, if a teller looked up several hundred such numbers in the last half hour of a shift, this could be a cause for concern. In this case the application would signal an immediate alarm to a supervisor

which would enable intervention. The underlying principles are the same: define clusters characterizing normal behavior and then look for outliers.

14.9 Notes and Sources

14.2. A good introduction to the broad field of fraud detection in health care is the paper [83].

14.4. The method of Condorcet Clustering described here is presented in the text book by Stéphane Tufféry [142]. See also the original paper by Michaud and Marcotorchino [89].

14.6. Often companies producing packages to combat fraud keep the details of their methods proprietary. For information on the specific approach described in this chapter, see Anderson et al. [4]. IBM's FAMS package has been installed at over forty healthcare insurance companies.