A Statistically Based System for Prioritizing Information Exploration Under Uncertainty *

Paul Helman
Department of Computer Science
University of New Mexico
Albuquerque, NM 87131
Jessie Bhangoo
Department of Computer Science
The University of Arizona
Tucson, AZ 85721-0077

August 29, 1996

Abstract

This paper examines the problem of prioritizing actions under uncertainty. Our motivating applications come from the domain of **data mining**. Data mining problems present the user with a huge collection of individual items (e.g., abstracts, medical histories, computer users' command histories) and require that these items be prioritized according to which should be pursued thoroughly. More precisely, each data item is assumed to be generated by one of two processes: A large majority of the data comes from a common, **mundane** process and a very small fraction comes from a rare, **phenomenon** process. The problem is to rank the information so as to optimaly direct the user in his or her pursuit of the data items that were generated by the phenomenon process.

*This work is supported in part by grants from Los Alamos National Laboratory and the National Security Agency.

Our previous work has developed the theoretical foundations of the information prioritization problem. The current paper summarizes these foundations, derives new theoretical results, and details initial experimental results of a prioritization system based on the theory. We focus here on **feature selection techniques** and the method of **model surrogates**, each tailored to the classes of prioritization applications of greatest current interest. Our results demonstrate the effectiveness of the techniques and motivate further research to improve the existing system.

1 Introduction

The problem considered in this paper is described abstractly as follows. We are presented with a large number of data points, each of which requires a binary decision (e.g., pursue the data point or discard the data point). Our problem is to **prioritize** (i.e., rank) the data points. The interpretation of the prioritization is that those points given the highest prioritization elicit one decision (e.g., pursue), while those given the lowest prioritization elicit the other decision (e.g., discard). By providing a prioritization, rather than simply a classification, the boundary separating the items eliciting each decision is dynamic; one can pursue the highest ranked items until time, money, or interest is exhausted, and the remaining items implicitly are discarded. The problem generalizes easily to nonbinary decisions (e.g., decisions which specify how rigorously a point is to be pursued).

Our motivating applications come from the domain of **data mining**. Here, we are presented with a huge quantity of information, of which only a small fraction is of particular interest. Since it often is costly to determine whether an individual item truly is interesting (e.g., by investigating the source of that information), resource limitations require that we prioritize the information so that the few items of interest are the ones we choose to pursue. To date, the primary application of our theory and software system has been **computer intrusion detection**, where the data to be mined is a computer audit trail and the goal is to prioritize for the computer security officer the audit trail's transactions (i.e., the records of user commands) according to the order in which they should be investigated. The concrete examples presented in this paper are drawn from the intrusion detection domain.

To simplify the presentation, we assume that all data points are generated by one of two stationary, stochastic processes: P, the interesting or **phenomenon** process and N, the **mundane** (or normal) process. Our results can be generalized to $n > 2$ processes. The data points are referred to as **transactions**, each of which assigns values to a common collection of **features**, i.e., a transaction is a feature vector. A **phenomenon detector** scores individual transactions based on their feature values; these scores induce a prioritization on the space S of all transactions. Once the transactions have been prioritized, a user may pursue the transaction from highest to lowest score, terminating the process when resources are exhausted, when enough interesting transactions have been pursued, or when the most recently pursued transactions have proven to be uninteresting, indicating that the mundane portion of the ranking has likely been reached.

The value of such a prioritization system is ever increasing in the context of today's information explosion, where an analyst often is flooded with an overwhelming amount of information and must make snap decisions under uncertainty. Important application areas include computer security, computer networks, telecommunications, radio astronomy, and biomedical testing. We believe that an automated decision support system based on the methods described in this paper can be a great aid to the human's decision process.

The theory and its realization in our software system address several aspects of the prioritization problem: feature selection (both the search for, and evaluation of, candidate feature sets), the role of generic models as surrogates for one or more of the processes, the specification and analysis of different classes of prioritization algorithms, issues of computational complexity, and comparisons with traditional classification. The current paper focuses on the feature selection and surrogate model aspects of the problem.

Though feature selection is a well-studied task in pattern recognition and other classification applications [2, 3, 4, 6, 12, 14, 15], the context of prioritization and the constraints of the applications of most interest to us mandate novel approaches. In particular:

1. We do not permit the evaluation of candidate feature sets by traditional error assessment against

sample evaluation data. This restriction is motivated by: (a) Applications in which samples of one of the processes (e.g., the phenomenon process) is unavailable, and (b) Our desire to employ search algorithms which consider a large number of candidate feature sets, thus requiring a faster evaluation of each candidate than would be supported by the traditional error assessment approach were restriction (a) relaxed.

2. As a result of (1), we employ a new class of feature set evaluation measures, and generic models to act as surrogates for the unrepresented process(es).

3. Performance of a detector, once built and run against actual data, is assessed in terms of the detector's **prioritization error**, rather than simple misclassification. In brief, prioritization error measures how much misdirection the ranking imposes on the user. Section 2 develops the mathematical definition of prioritization error and summarizes several theoretical results. As will be discussed, optimal prioritization is a significantly more difficult task than is optimal classification, in the sense of P vs. NP (i.e., the class of polynomial time algorithms vs. the class of nondeterministic polynomial time algorithms [5]).

The remainder of this paper is organized as follows. Section 2 gives a brief summary of the theoretical foundations of our detection methodology and of our main theoretical results. Appendix 1 summarizes the notation and terminology introduced in Section 2 and used throughout the paper. Section 3 studies our predictive evaluation measures, an essential component of our system and a main subject of the experiments. Section 4 briefly describes the software system implementing our methodology, while Section 5 describes the experiments performed with our system and offers interpretations of the results. Appendices 3 and 4 supply details of these experiments. Finally, Section 6 discusses some possible future directions.

2 Summary of Theoretical Foundations

A detailed introduction to the problem considered in this paper and its theoretical foundations can be found in [8, 9]. This section provides only an overview of the material contained there.

2.1 Theoretical Results

We model transactions as generated by two, independent, stationary stochastic processes, the **mundane** process N, and the **phenomenon** (interesting) process P, each with its image in the finite transaction space S. A training sample DB generated by one or both processes is assumed available, and RS denotes the stream of transactions, generated by a mix of the two processes, which must be prioritized. (More precisely, each of DB and RS is a sequence of transactions from S, with possible repition of transactions within each sequence.) In the current paper, we consider the most difficult problem version in which *DB is generated by only one (e.g., the mundane) process.* This is motivated by applications, such as computer intrusion detection and radio astronomy, where it is believed that the data points which will be of most interest often will be generated by a previously unknown process (e.g., a new intrusion technique, an unknown intelligence). The theoretical results presented in this section apply also, with appropriate modification, to scenarios in which a training DB consisting of samples of both processes is available. In this case, of course, techniques for exploiting samples of the second process enable more accurate prioritization than is possible under the scenario of current interest.

We consider two classes of detectors: Both classes train on DB. $Class\ I$ defines real-time detectors that must evaluate a transaction as it is encountered, independent of other transactions appearing in RS, and $Class\ II$ defines detectors that may examine all transactions in RS before evaluating them. Naturally, $Class\ II$ contains $Class\ I$ as a special case. $Class\ I$ detectors are used when time and/or space constraints are too severe to allow a comparative evaluation of the transactions[9].

More formal definitions of the two classes of detectors appear in Appendix 2.

In order to measure the quality of a detector, we introduce notions of loss and risk. Loss is based on prioritization error as defined in [8]. For any phenomenon detector pd, training set DB, and rank stream RS, $L(pd, DB, RS; \theta)$ is the prioritization error incurred by pd over all transactions pairs from RS when the sample is DB and $\theta = (\theta_N, \theta_P)$ is the true process pair's distribution. See Appendix 2 for a complete definition of loss function L.

From the definition of Bayes risk we have

$$rsk(pd) = \int_{\Theta} R_{pd}(\theta) g(\theta) d\theta$$

where Θ denotes the space of all possible process pairs θ, $g(.)$ is a prior density or probability function over Θ, and

$$R_{pd}(\theta) = \sum_{DB,RS} L(pd, DB, RS; \theta) Pr\{DB, RS|\theta\}$$

In many applications, $g(.)$ will be a discrete probability function, in which case the integral is interpreted as a sum.

An optimal detector pd^* is a detector minimizing $rsk(pd)$. In practice, most detection problems are difficult to solve optimaly. Helman[9] demonstrates several theoretical differences between prioritization problems and traditional classification problems. These results include:

(*i*) Optimal $Class\ II$ prioritization is completely consistent with optimal $Class\ II$ classification (that is, when the classifier may inspect all the information before classifying any of the information pieces), while optimal $Class\ I$ prioritization is not always consistent with optimal $Class\ I$ classification (that is, when the classifier must classify each information piece as it is encountered). In particular, it may be optimal to rank x as less likely to be interesting than y, yet optimal also to classify y as mundane and x as interesting. Hence, it need not be the case that an optimal ranking can be turned into an optimal classification with appropriate choice of cutoff point.

(ii) Optimal $Class\ II$ prioritization, and both versions of the classification problem, are solvable in polynomial time, assuming the posterior

$$Pr\{P|x; DB, RS\} = \int_{\Theta} Pr\{P|x; \theta\} * f(\theta|x, RS, DB)d\theta$$

,

is polynomial time evaluable. ($f(\theta|x, RS, DB)$ denotes the posterior distribution on Θ, given transaction x, training sample DB, and rank stream RS.) In contrast, under the same assumption (with the posterior conditioned on x and DB but not RS), optimal $Class\ I$ prioritization is not solvable in polynomial time, unless $P = NP$. (As is standard in complexity theory, P denotes the class of polynomial time solvable problems, and NP denotes the class of nondeterministic polynomial time solvable problems. It is widely accepted that $P \neq NP$. See [5] for an introduction to the theory.)

Our current focus is Class I detectors; the apparent intractability of optimal Class I prioritization motivates the heuristic approach explored in the remainder of this paper.

2.2 Heuristic Phenomena Detectors

To begin, we consider the simplest of detectors. The **naive detector** creates a model of mundane data based on frequency counts $count(DB, x)$ of the transactions in the training sample DB (i.e., the transactions' histogram for DB), evaluating as most interesting those transactions with the lowest counts. In particular, the naive detector prioritizes transactions according to the simplistic estimate $r(x) = m_U(x)/n_f(x)$ of their **likelihood ratios** where, for each $x \in S$:

$n_f(x) = count(DB, x)/|DB|$, which estimates the probability $n(x) = Pr\{x|N\}$ that the mundane process generates x, and

$m_U(x) = 1/|S|$, which substitutes for an estimate of the probability $m(x) = Pr\{x|P\}$ that the phenomenon process generates x, by treating all transactions as equally likely under P.

The naive detector is consistent with **anomaly detection**, as it equates level of interest with rarity [8]. In our framework, anomaly detection is treated as a special case of phenomena detection, one obtained by using for the phenomenon model the **uniform surrogate**, in which $m_U(.)$ is constant over S. See Section 2.4 and [8, 9] for more on the relationship between anomaly detection and choice of surrogate model.

The naive detector, while satisfactory for some applications, has two significant limitations which make it inappropriate for many of the most practically important prioritization problems.

- When the transaction space S is large relative to the training sample DB, the frequency counts will be low and provide unreliable estimates of true probabilities. In particular, in many applications, all transactions appearing in DB will have count 1 (so called **singleton** transactions) and most transactions appearing in RS will not have appeared at all in DB. Consequently, any detector based on raw transaction frequency counts will have no legitimate basis by which to prioritize the data points.
- Rareness is equated *de facto* with interest. In practice, it often may be appropriate to prioritize based on other relationships between the processes N and P. Section 2.4 illustrates this point.

The detectors considered in the current paper remain within the basic framework described for the naive detector, prioritizing according to estimates of the transactions' likelihood ratios $r(x)$. We improve upon the naive detector's prioritization by employing two techniques, equivalence classing and generic model surrogates. The former technique improves upon the naive detector's raw frequency count estimate $n_f(x)$ for $n(x)$, while the latter permits alternative model surrogates to replace, when appropriate, the naive detector's uniform surrogate $m_U(x)$.

The techniques of equivalence classing and model surrogates are summarized in the following two subsections. The remainder of the paper studies the techniques in detail, both theoretically and experimentally.

2.3 Equivalence Classing

Equivalence classing is a transform of S that aggregates into **transaction classes** distinct transactions of S. The transaction classes can be induced by any equivalence relation on (or, equivalently, by any partition of) S. Transactions falling into the same class are treated as indistinguishable entities. The primary motivation for such a transform is that, in general, frequency counts of transaction classes more accurately reflect class probabilities than do frequency counts of raw transactions represent raw transaction probabilities. This is a simple consequence of increasing the representation in the sample DB of the events (transaction classes rather than raw transactions) whose probabilities are being estimated. Of course, in order that such transformations be useful in prioritization, some relationship must exist between class probabilities and the probabilities of the individual transactions in the class. A theoretical analysis of *a priori* densities on process distributions supporting equivalence classing appears in [9]; the predictive evaluation measures introduced in the current paper's Section 3 are derived from these *a priori* densities.

In theory, any equivalence relation on S may be considered. In practice, **value clustering** is by far the transform of S most often appropriate. Value clustering aggregates into clusters distinct feature values, with values falling into the same cluster treated as indistinguishable. This transform is useful if, for example, a feature value is taken from a continuous (e.g., floating point) domain, if the features represent values recorded in semantically useless precision (e.g., time of day measured in milliseconds), or if collections of values are meaningfully analyzed together (e.g., computer user classes, each consisting of users who are somehow associated with one another).

An important limiting case of value clustering collapses into a single cluster all the values of a feature's domain. Since all transactions will agree on the collapsed feature, this is equivalent to eliminating the feature. That is, *feature selection may be treated as a special case of value clustering*. The motivation for feature selection (which we refer to also as **feature projection**) is that often only a subset of the full feature set actually contributes to differentiating the processes. In this case, ignoring the remaining features increases the accuracy of the probability estimates provided by the sample. Techniques for feature selection, specifically tailored for the context of our prioritization problem, are a focus of the remainder of the paper.

The following notation and terminology is used in connection with feature selection. Π denotes any set of features. For $x \in S$, $[x]_\Pi$ is the equivalence class consisting of all $y \in S$ agreeing with x on each feature in Π. If T is a sample (i.e., sequence) of transactions, $count(T, [x]_\Pi)$ denotes the sum of the counts in T of all transactions in $[x]_\Pi$. When the training sample DB is understood, $n_\Pi(x)$ will denote $count(DB, [x]_\Pi)/|DB|$.

2.4 Surrogates

Equivalence classing transforms improve upon the naive detector's estimate of the denominator $n(x) = Pr\{x|N\}$ of $r(x)$. But what of the numerator, $m(x) = Pr\{x|P\}$, the probability that the phenomenon process P generates transaction x? As already noted, the naive detector employs the uniform surrogate which treats all transactions as being equally likely under P. In our research, we have considered several alternative surrogates. In this paper, in addition to the uniform surrogate, we consider the **independence** surrogate. The independence surrogate treats all features as if they were statistically independent, with the marginal distributions based on the frequencies observed in DB. The independence model helps identify transaction's whose feature values are significantly rarer in combination than are their individual marginal frequencies [8, 9]. The following example compares in an intrusion detection setting the effect of using the independence and uniform surrogates.

Example: Suppose that transactions consist of the features *User* and *Command*, with respective value sets {Fred, Sue} and {execute, edit}. Suppose further that the training sample DB consists of 100 transactions, yielding observed frequencies of the four possible transactions as shown below.

freq(Fred, execute) = 4   freq(Fred, edit) = 90

freq(Sue, execute) = 2   freq(Sue, edit) = 4

Since the raw frequency counts are sufficiently large, we do not eliminate features and estimate, for example, θ_N(<Fred, execute>) = .0400 and θ_N(<Sue, execute>) = .0200.

When the uniform model θ_U is used for the misuse surrogate, we obtain for any $x \in S$ $\theta_U(x)$ = .250, and hence pd_U<Fred, execute> = 6.25 and pd_U<Sue, execute> = 12.5. Therefore, under the uniform misuse surrogate, <Sue,execute> is deemed to be the most suspect transaction simply because it is the least frequent in the sample, a result consistent with anomaly detection applications.

The independence surrogate θ_I assigns to each transaction

$$x =< A_1 = v_1, ..., A_i = v_i, ..., A_n = v_n >$$

the value $\theta_I(x) = \prod_{i=1}^{n} marginal(v_i)$, where we estimate the marginal probability as $marginal(v_i) = \sum_{x \ni x[A_i]=v_i} count(DB, x)/|DB|$. Hence, in our example,

θ_I(<Fred,execute>) = (.940)*(.0600) = .0564

θ_I(<Sue,execute>) = (.0600)*(.0600) = .00360

pd_I(<Fred, execute>) = 1.41; pd_I<Sue, execute> = 0.180

Therefore, under the independence surrogate, <Fred,execute> is deemed the most suspicious transaction because Fred is doing something *relatively unusual, for him*.

It must be noted that no claim is made that the surrogate models resemble the actual phenomenon processes. Rather, the hope is that these simple surrogates will abstract some significant characteristics that *differentiate* the mundane and phenomenon processes.

2.5 Constructing the Detector

Given the detection framework outlined in the previous section, the following summarizes the construction tasks at hand.

- Choose an equivalence classing. In this paper, we restrict attention to the classings induced by feature subsets; in particular, we study methods for evaluating candidate feature sets. Since we do not permit traditional error assessment (as was motivated in Section 1), novel evaluation methods must be employed.
- Choose a surrogate. In this paper, we compare the effects of using the uniform and independence surrogates.

The result of detector construction can be viewed as a pair $(surrogate, \Pi)$ which induces the detector $pd(x) = m_{surr}(x)/n_{\Pi}(x)$. A promising extension of this framework is to construct a k-tuple $< (surr, \Pi)_j >$ of pairs inducing a k-tuple $< pd_j(x) >$ of ratios, along with a method of synthesizing the j evaluations. While the current paper considers only simple (i.e., dimension $k = 1$) detectors, a future paper will report on the behavior of k-ary detectors, for $k > 1$.

3 Predictive Measurement Functions

As discussed in the previous section, a candidate detector is specified by a feature subset Π and a phenomenon process surrogate. As we search the space of detectors, the effectiveness of a given candidate detector is predicted by an effectiveness measurement function. This measure is *predictive* in that it evaluates a candidate detector against the training data, then reports how well the detector, in the future, will be able to distinguish mundane data from interesting data. In traditional supervised learning, the training data generally consists of samples of all the categories to be recognized and a classifier's assessment is based on rate of error on a portion of the sample reserved for evaluation. In the context of the applications considered in this paper, the detector evaluation problem is particularly daunting because the measure must evaluate a detector based on training data that is *entirely mundane*, yet must be able to predict the detector's ability to prioritize data streams consisting of a mix of mundane and interesting items.

Many such evaluation measures are possible, though, of course, no single technique can evaluate accurately detectors for all possible phenomena which may be encountered. For example, a simple evaluation measure, based on minimizing the number of singleton classes (i.e., classes with one occurrence) which result when the training database is projected onto Π was proposed and studied [8]. This singleton measure is motivated by Good's estimate [7] of the mass attributable to that fraction of the transaction space unrepresented in the training sample. Subsequently, we performed a detailed analysis of the risk expression the detector is to minimize, leading us to several more sophisticated evaluation measures. In this section, we present definitions and a brief theoretical analysis of the most promising of these evaluation measures (the reader is referred to [9] for a more detailed development of these measures). Section 5 of the current paper reports on preliminary experimental results obtained with one family of these measures. These initial results suggest the measures work well for a wide range of applications.

Basing a detector on the counts $count(DB, [x]_\Pi)$ can be viewed formally as an attempt to rank the transactions correctly relative to the hypothesis H_Π that only the features in Π are relevant to distinguishing the processes. Our evaluation measures attempt to select the subset Π to be used by determining the H_Π which, if true, would lead to the largest decrease in risk incurred when a detector orders each pair of transactions optimally, rather than suboptimally, with respect to H_Π. This evaluation with respect to H_Π can be weighted by $Pr\{H_\Pi | DB\}$, the probability that hypothesis H_Π is true, given the training data DB.

The following notation and terminology is used in the description of the measures. Distribution θ on S is **consistent with** hypothesis H_Π if $y \in [x]_\Pi$ implies $\theta(x) = \theta(y)$. Distribution $\theta_{N,\Pi}$ defined by $\theta_{N,\Pi}(x) = count(DB, [x]_\Pi)/(|DB| * |[x]_\Pi|)$ maximizes $Pr\{DB|\theta_N\}$, among distributions θ_N that are consistent with hypothesis H_Π. $\theta_{P,\Pi}$ will denote the surrogate phenomenon distribution (e.g., the uniform or independence surrogate) we have chosen to employ.

Ideally, an evaluation measure accounts for the magnitude of the difference $\theta_{P,\Pi}(x)\theta_{N,\Pi}(y) - \theta_{P,\Pi}(y)\theta_{N,\Pi}(x)$, averaged over all pairs x, y of transactions (which represents the importance of the optimal orderings if H_Π were true), weighted by $Pr\{DB|\theta_{N,\Pi}\}g(\theta_{N,\Pi}, \theta_{P,\Pi})$ (which represents how likely is H_Π given DB and prior distribution g over possible distribution pairings). Such a measure can be formalized in terms of the risk expression to be minimized [9].

Based on this ideal, we define a family of evaluation measures which, collectively, we refer to as the θ-**separation measures**.

$$M_1(\Pi) = \sum_{\{x,y\}\ni[x]_\Pi \ and \ [y]_\Pi \ in \ DB} |\theta_{P,\Pi}(x)\theta_{N,\Pi}(y) - \theta_{P,\Pi}(y)\theta_{N,\Pi}(x)| \, \mathrm{Pr}\{DB|\theta_{N,\Pi}\}g(\theta_{N,\Pi}, \theta_{P,\Pi})$$

$$M_2(\Pi) = \sum_{\{x,y\}\ni[x]_\Pi \ and \ [y]_\Pi \ in \ DB} |\theta_{P,\Pi}(x)\theta_{N,\Pi}(y) - \theta_{P,\Pi}(y)\theta_{N,\Pi}(x)|$$

$$M_3(\Pi) = \sum_{\{x,y\}\ni[x]_\Pi \ and \ [y]_\Pi \ in \ DB} |\theta_{P,\Pi}(x)count(DB, [y]_\Pi) - \theta_{P,\Pi}(y)count(DB, [x]_\Pi)|$$

The larger these sums for Π relative to that for other feature sets, the more significance we attach to Π.

Measure M_1 is motivated by the preceding discussion. M_2 drops the posterior probability factor for H_Π, treating all H_Π as equally likely, regardless of the training data. This often is necessitated by the difficulty of assigning appropriate prior distributions, and has the effect of a g that favors H_Π for small Π. M_3, by not normalizing class counts by class cardinality, further favors H_Π for small Π.

The experiments of Section 5 assess the performance of four variants of measure M_2: (*i*) the measure calculated over all transaction pairs $\{x, y\}$ such that $count(DB, [x]_\Pi) > 0$ and $count(DB, [y]_\Pi) > 0$ vs. (smaller) random samples of transaction pairs (as is necessitated when the number of pairs is large) and (*ii*) the exact and sampled measures using each of the independence surrogate and the uniform surrogate.

Whatever evaluation measure we select, we are confronted with a difficult search problem: search the space of feature subsets for a subset Π which scores best under the measure. In [8] we demonstrate that finding the subset Π which optimizes the singleton reduction measure is an NP-hard problem (referred to as SRP, for the singleton reduction problem), and further conjectured that the NP-hardness of more complex and useful measures would follow from the NP-hardness of singleton reduction. We establish now that, indeed, the NP-hardness of optimizing the θ-separation measures follows from that of the singleton reduction problem.

Define the class count spread maximization problem (CSMP) as follows.

INSTANCE: Database DB of some finite number of transactions, each consisting of D features, and a positive integer B.

QUESTION: Is there a subset Π of features such that the pairwise class spread (as given by M_3 above, with $\theta_{P,\Pi}(x)$ identically 1 for each Π) is B or greater?

The following theorem demonstrates this decision problem to be NP-complete. The NP-hardness of the corresponding optimization problem then follows immediately.

Theorem: CSMP is NP-complete.

Proof: See Appendix 2.

Corollary 1: CSMP with an explicit constraint on the subset cardinality is NP-complete.

Proof: See Appendix 2.

Corollary 2: CSMP is NP-complete if class counts are divided by class cardinality, as in measure M_2. This result holds for both the cardinality-constrained and unconstrained versions of the problem.

Proof: See Appendix 2.

4 Experimental System

In this section, we describe our software system implementing the previously described theory. The section which follows reports a small suite of experimental results which, we believe, is encouraging enough to justify further research and development along the lines of our theory.

Our experimental system performs the following functions:

- Input parameters to configure the system for an experimental run. The parameters include a choice of process pair (N, P).
- Generate a training sample DB of mundane data, in accordance with the mundane process N.
- Use the techniques described in the previous sections to construct a Class I detector pd.
- Generate a rank stream RS as a blend of processes N and P. Then use pd to rank RS and assess the error incurred by this ranking.

The following subsections elaborate on these system functions.

4.1 Input Parameters

The following parameters are varied in the experiments reported in the following section. Additional parameters are available, but not discussed here.

1. Number n of features.
2. Domain size D of each features.
3. Description of the mundane process, N (see section 4.2).

4. Description of the phenomenon process, P (see section 4.2).

5. Size of DB (the training sample).

6. Phenomena surrogate model (see section 4.3).

7. λ, the fraction of test points in rank stream RS generated by the mundane processes.

8. Number of ranks streams over which to average results.

4.2 Data Generation

As noted in the previous section, input parameters specify the mundane (N) and phenomenon (P) processes. The system supports an (ever growing) database of statistical processes from which to choose. In brief, a process is defined by specifying relationships between feature distributions (e.g., all features independent, a statistical dependency between certain groups of features) and the marginal distributions of the independent features.

Once N has been specified, the training data DB is generated in accordance with its probability distribution. After the detector is constructed by training on DB (see Section 4.3), the rank streams are generated as a blend (with fraction of mundane determined by λ) of transactions from N and P.

4.3 Detector Construction

The heart of the software system is the algorithms that construct detectors from the training data DB. The detector construction task was summarized in Section 2.5, and explored further in Section 3.

The primary goal of the experiments reported in this paper is to justify two components of our approach: the predictive measures (described in Section 3) and the method of surrogates (described in Section 2.4). As such, the system has been configured to run in an experimental mode, with the following special behaviors.

- Equivalence classings are restricted to those induced by feature subsets.
- Four different measures are considered, as described in Section 3. Two of the measures utilize the uniform surrogate and two utilize the independence surrogate. A detector is constructed for each measure, based on the feature subset scoring best on that measure. The quality of these detectors is compared, as described in Section 4.4.
- In order to factor our the effect of different search algorithms (in particular, which candidate feature sets are evaluated), an exhaustive search is performed. This necessitates that we limit the number of features considered. Of course, for an actual application (where the number of features often is in the hundreds), exhaustive search is not feasible and sophisticated, heuristic search techniques must be employed. A study of such heuristics is a focus of current research.

4.4 Detector Assessment

Each of the detectors induced by the measures is used to prioritize each of the generated rank streams. Additionally, as benchmarks, the naive detector (based on the full feature set; see Section 2.2) and the theoretically optimal detector for the known, fixed process pair (N, P), are used to prioritize the rank streams. As identified in [8], the optimal pd for known (N, P) is given by the true likelihood ratio $pd(x) = Pr\{x|P\}/Pr\{x|N\}$. Of course, this detector is of theoretical interest only, since its computation requires perfect knowledge of the process distributions. Note that since the images of the processes may overlap, even the optimal detector will generally incur some non-zero error.

The **observed prioritization error**, averaged over the generated rank streams, is then computed for each detector, allowing us to assess the relative qualities of the predictive measures on which the detectors were based. The observed prioritization error for a detector on a rank stream with test points of known origin (i.e., the test points are marked N or P) is simply the number of pairs of points p_i, p_j that the detector ranks as $p_i < p_j$, yet p_j is marked P and p_i is marked N. (Additionally, if a mundane test point and a phenomenon test point are ranked as equal, an error of one half is incurred.) This observed prioritization error is the analogue of loss and risk (see Section 2) which is appropriate when analyzing a fixed rank stream whose points are of known origin.

5 Experimental Results

This section presents the results of two sets of experiments conducted with our software system. As was noted earlier, the primary goals of these experiments were to lend validity to our predictive measurement functions and to assess the effect of the choice between the uniform and independence surrogates.

The first set of experiments was based on simple, "generic" mundane and phenomenon model pairs. The second set uses models derived from the generic models, but modified slightly to reflect more closely classic intrusion detection scenarios.

Before describing the models employed and presenting our results, we describe briefly the experimental design methodology that was employed to arrive at the set of experiments that we conducted.

5.1 Experiment Design

For this problem, experiment design consists largely of limiting the number of test runs. Due to the large number of possible input parameter combinations, we employee two techniques to reduce the total number of tests.

Parameter Domain Reduction

An experimental problem occurs when an input parameter can have a large number of values (called **levels**). Testing each possible input value, or even a periodic sample will result in an excessive number of experiments — too many to be feasible. To solve this problem for our purposes, assume that there is a linear relationship between a parameter's value and its effect on the performance of a detector. This way, only two experimental input levels need to be chosen for testing[1]. Naturally, this reduces the accuracy of the extrapolated results, but since this is the first series of experiments, the first priority is to discover general trends (eg. positive correlation). For this goal, a linear model will suffice. We employed this technique in choosing values for the parameters specifying the number of features and their domain sizes, and in analyzing their effects on the results.

Parameter Space Reduction

Consider two logically related parameters. Sometimes it is not necessary or too costly to test every possible combination of settings of these parameters. When this is the case, choose logical "pairs" of parameter values to test. For example, the parameters *Mundane Process Model* and *Mundane Process Marginal Distribution* are logically related. Combine these parameters into one single parameter, *Mundane Process*, consisting of representative $(Model, Distribution)$ pairs. This approach can be applied to the parameters *Phenomena Process Model* and *Phenomena Process Marginal Distribution* in the same way to create *Phenomena Process*. Continuing this same method, these two new parameters may now be combined into one single parameter, the (N, P) process pair, by considering only pairs of mundane and phenomenon processes that appear to be potentially interesting.

5.2 Experimental Models

Generic Models

Two models were developed to represent general information domains and how phenomena may occur within them. Experiments were conducted on these models to demonstrate the basic capabilities of the system, in particular, to assess the quality of the feature set evaluation measures, and the role of model surrogates. The (N, P) model pairs constructed for each experiment is such that only the first two features, A_1 and A_2, are relevant to distinguishing the processes. The ranking

[1]This is done for only parameters that take their values from an *ordered series* as opposed to parameters that possess no logical ordering of their values (*eg.* choice of surrogate function).

produced by a detector based on a feature set containing the irrelevant attributes can be expected to incur a larger prioritization error than a ranking based on feature sets containing only A_1 and A_2. Hence, a feature set evaluation measure ideally will give its highest evaluations to sets containing only these features, and our proposed measures should be assessed with respect to how closely they approximate this ideal behavior.

Experiment 1: Experiment 1 was intended primarily to test our measures' feature selection capabilities. Model P is chosen so that the features are pairwise independent, with each marginal probability distributions uniform over the domain. N is chosen so that the features are pairwise independent and the marginal probability distributions of the first two features, A_1 and A_2, are skewed to favor a few certain values more than the others (see Appendix 3 for the detailed marginal distributions used in this experiment). The remaining features are uniformly distributed over the domain.

Experiment 2: Experiment 2 was intended to test not only feature selection capabilities, but also the effect of surrogates. To this end, Experiment 2 includes a "crossover" relationship between N and P. In N, feature A_2 is dependent on feature A_1 with all other features pairwise independent. The correlation between the features is as follows: If A_1 is odd (even), A_2 is biased to a low (high) value over the domain. The marginal probabilities of the independent features are binomially distributed over the domain, with distribution parameters $p = 0.5$ and $N = D$ (domain size) (see Appendix 3 for the detailed marginal distributions used in this experiment). This was chosen as a discrete analog of a Gaussian distribution. P is chosen so that A_2 is dependent on feature A_1 with all other features pairwise independent. The correlation between A_1 and A_2 is the reverse of that in the N model (that is, A_2 is biased high when A_1 is odd and vice versa). The independent features are binomially distributed over the domain as in N.

To implement the "bias" in Experiment 2, two values were randomly selected from a uniform distribution over the domain. The minimum (maximum) value is selected to bias the result low (high).

Four parameters were varied in each experiment. The domain size was tested with $D = 4$ and

$D = 10$. The number of features was tested with $n = 3$ and $n = 5$. Finally, the uniform and independence surrogates were utilized in both the feature evaluation measures and the detectors themselves.

The remaining parameters are set as follows: The size of DB is 1000. The number of rank streams per run is set to 100 to ensure a good average random distribution and reduce the effects of experimental "noise." The size of each RS is fixed at 100 and λ is set to 0.7. λ can be set lower than what we would expect in reality because this value does not affect the optimal ranking. Setting the value low eases testing by providing more sample points in the test data that come from the P process.

Computer Intrusion Models

The generic models employed in the first set of experiments were designed to abstract several of the types of difference between the mundane and phenomenon process that are expected to characterize many application domains. We consider now the intrusion detection domain in particular, deriving from the generics three model pairs that are somewhat more reflective of that domain. Since these models less resemble than do the generics the two simple surrogates our detection system employs, a more difficult test of our methodology is obtained. In particular, these experiments provide a more challenging test of our premise that simple surrogates can serve to distinguish typical mundane and phenomenon process pairs, even when those processes bear little direct resemblance to the surrogates.

We begin with a brief description of computer intrusion detection, an important application of phenomena detection[8]. A System Security Officer (SSO) often has the job of studying logs of computer transactions (i.e., **audit trails**) for possible misuse or abuse. Since the volume of computer activity on even a moderately busy computer system easily can overwhelm the SSO, an automated system for prioritizing the audit trail's transactions is of great potential value. Once the audit trail is prioritized, the SSO pursues the transactions from highest to lowest priority, until time, money, or interest is exhausted.

In 1980, James P. Anderson motivated the development of intrusion detection systems by describing different threats to computer systems [1]. Three of these threats could be detected by intrusion detection but not system access control, the traditional approach to computer security.

> Misfeasors — valid users who misuse their access to the system
>
> Masqueraders — malicious outsiders have somehow obtained a valid user's ID and password
>
> Clandestine users — valid users who have breached internal access control and auditing procedures

As further validation of the methods presented in this paper, we developed models of N (mundane system use) and P (computer misuse or abuse) that are based on the above three scenarios and performed three corresponding experiments. These models are derived from the previously described generics, but more closely reflect the intrusion scenarios. As in the previously described experiments, our system employed both the uniform and independence surrogates, for both the feature evaluation measures and the detectors themselves.

Transactions consist of five features. Because this is a specific application, semantic meaning is assigned to the features, and the domain sizes are no longer homogeneous. The features are described as follows, with domain sizes in parentheses: $A_1 = User$ (10), $A_2 = Command$ (10 for the misfeasor scenario, 33 for the masquerader and clandestine scenarios), $A_3 = Time\ of\ Day$ (100), $A_4 = Terminal$ (20), and $A_5 = System\ Load$ (100). Notice that $Time\ of\ Day$ has been pre-clustered into a set of time regions, each 14.4 minutes long. This is an example of **value clustering** as described in Section 2.3 and [8].

As was the case with the generic models, only two features ($User$ and $Command$ in the current experiments) will be relevant to distinguishing the processes. The ranking produced by a detector based on a feature set containing the irrelevant attributes can be expected to incur a larger prioritization error than a ranking based on feature sets containing only $User$ and $Command$. Hence, as in the first set of experiments, a feature set evaluation measure ideally will give its highest evaluations to sets containing only these features, and our proposed measures should be assessed with respect to how closely they approximate this ideal behavior.

Experiment 1 (misfeasors) derives its N and P models from the generic N and P of the first experiment of the previous section. In particular, N is defined as follows. $Command$ is exactly correlated with $User$ – each user performs exactly one command, unique to that user. $User$ (and hence $Command$) is biased away from one particular value. The remaining $User$ values (and hence $Commands$) are selected with approximately equal probability, and the remaining features are statistically independent and uniformly distributed. Model P differs from N only in that $User$ is biased *toward* the user who is rare in N. (See Appendix 3 for more details of the distributions.) In this experiment, $Command$ can be thought of simply as a "trace" that a user has accessed the system, and the (N,P) pair represents an intrusion scenario in which a particular user overuses the system; the remaining features are not explicitly affected by the intrusion.

Experiment 2 (masquerader) derives its N and P models from the generic N and P of the second experiment of the previous section. In particular, N is defined as follows. All features, except $User$ and $Command$, are pairwise independent and binomially distributed. $Command$ is correlated with *classes* of $Users$. That is, we divide the users into two classes, each of which is more likely to use a different subset of commands than is the other user class. P has a crossover relationship between $Command$ and $User$ classes that opposes the relationship in N. That is, each class of user is more likely in P to use the subset of commands that it is less likely in N to use. (See Appendix 3 for more details of the distributions.) This pair (N,P) represents an intrusion scenario in which one or more users behaves like other users.

Experiment 3 (clandestine) employs a somewhat more challenging version of the pair (N,P) employed in Experiment 2. Here, in P, the relationship between $User$ and $Command$ exists as in Experiment 2, but in N there is independence between all features, including $User$ and $Command$. (See Appendix 3 for more details of the distributions.) This pair (N,P) represents an intrusion scenario in which there is an overuse of certain commands by certain users, a situation that does not exist in N, perhaps due to administrative restrictions.

5.3 Results and Interpretation

Generic Models

In Experiment 1, the detectors based on the feature subset $\{A_1, A_2\}$ and the uniform and independent surrogates are the clear preference of the θ-separation measures. Comparing these detectors to the naive detector (which does no subset projection) and the theoretically optimal detector (which has perfect *a priori* knowledge of the N and P distributions), we find that they perform noticeably better than the naive detector and almost as well as the optimal detector. Additionally, the $\{A_1, A_2\}$ detectors clearly perform better than typical subset detectors. The quantitative experimental results are given in Appendix 4.

These simple result lead to two initial conclusions. First, eliminating irrelevant features improved the performance of detectors significantly. Second, our θ-separation measures were able to identify which features are relevant. The first result, of course, was to be expected; the second provides support for our measures.

It is worth noting that domain size affected the quality of the measures. In the experiments where the domain size was decreased ($D = 4$), the measures favored larger subsets. These larger subsets, of course, did not perform as well as the smaller $\{A_1, A_2\}$ in practice. It is conjectured that the decreased data space when $D = 4$ reduces the advantage of subset projection because the raw, unprojected category counts are large enough to be meaningful themselves. That is, when the domains are larger, the sample is more representative of the true frequencies, and feature selection therefore appears to the measures to be of minimal value. It is difficult to verify this hypothesis by examining the results of the 3 feature experiments (which would also reduce the size of transaction space S) because with the reduced number of features, there is not much difference between the full set and the $\{A_1, A_2\}$ subset. The exponential nature of the current experiment design makes it difficult to use larger spaces to adequately test this hypothesis. This will be the subject of future testing.

Different combinations of surrogates were tried in the measures and the actual detectors. The best results were recorded when the same surrogate was used in both the measure and the detector. In general, though, the uniform surrogate resulted in lower absolute prioritization error than did the independence surrogate. This seems to be a logical result, given the regular nature of the N and P process models in this experiment. Globally rare transactions are exactly the phenomenon transactions. A direct comparison between the two surrogates is addressed in Experiment 2.

Experiment 2 demonstrates the different consequences of the surrogates, as well as the effect of dependencies between features. The features are binomially distributed; since the binomial distribution is a discrete analog of the Gaussian distribution, it represents a realistic distribution over the domain.

In all sample runs of the experiment, the selected subset detectors performed significantly better than the naive detector. However, as expected, the feature selection powers of the measures were weaker in this experiment than in the first. This is the result of the relationship between N and P. The predictive measures have knowledge of the processes only through the training sample, and this does not indicate the full significance of features A_1 and A_2; the full significance of these two features becomes apparent only after the P model is known as well. Consequently, the best our surrogates could hope to achieve is a kind of generic separation of the models and, indeed, one version of the θ-separation measure was able to identify the best subset detector $\{A_1, A_2\}$ in each run of Experiment 2.

When using the uniform surrogate, no detector performed as well as it did in the first experiment, both in absolute terms (*ie*. prioritization error) and relative terms (*ie*. compared to the optimal detector). The independence surrogate, however, enabled the measures to choose a better performing subset. When used in the actual detector, the independence surrogate enabled the detectors to perform better in absolute terms and also achieve close-to-optimal results. This is significant because previously developed systems, called **anomaly detection systems**, have implicitly relied on the uniform surrogate to model the phenomenon process (see section 2.2). This experiment demonstrates that anomaly detection may not be sufficient for even simple process models. As with Experiment 1, the experimental results are given in Appendix 4.

Computer Intrusion Models

Although these three experiments were more difficult tests than the generic models, the results were still promising. Even though none of the variants of the θ-separation measures described in Section 3 found the best detector in any of the experiments, often the favored detector came close to the best possible subset detector. Further, at least one measure was able to find a detector that significantly improved upon naive detection in every case. The quantitative numerical results for all experiments are given in Appendix 4.

In Experiment 1, two variants of θ-separation chose good detectors. These detectors gave a 34% reduction in prioritization error over naive detection. It was difficult to benchmark the results because optimal detection was very close to ideal (*ie*. very small error). Although θ-separation did not find the best subset detector (which gave over 95% improvement in prioritization error from naive detection), the detector it did find rated among the top 6 detectors out of 32 in all 100 test trials. In 96 of these trials, it rated among the top 4. It outperformed the naive detector in every trial.

In Experiment 2, the detectors chosen by θ-separation performed slightly better than average, noticeably better than the naive detector and gave only 1.19 times the error of the best subset detector when using the uniform surrogate. Using the independence surrogate gave an error that was 1.79 time that of the best subset detector (also using the independence surrogate). In absolute terms though, the error when using the independence surrogate was less than when using the uniform surrogate. In general, most subset detectors gave similar results, but the selected detectors distinguished themselves by consistently performing better than average and coming closer to optimal detection than the typical subset detector.

In Experiment 3, the uniform and independence surrogate gave similar results, with the uniform having only a slight advantage. As in Experiment 2, all chosen detectors gave similar results. The error was approximately 1.5 times that of the optimal detector in all cases. The difference was that the detectors chosen by θ-separation gave lower error than the naive detector in approximately half of the trials (51 times out of 100 trials), whereas a typical detector did not improve on naive detection at all (typically 0 times out of 100).

Conclusions

These experiments provide a proof of concept for the statistical phenomena detection approach proposed in [8, 9] and described further in this paper. In both sets of experiments, our predictive evaluation measures led us to good, and often optimal, feature subsets. In particular, *features irrelevant to distinguishing the processes were almost always absent from feature sets favored by these measures*. Further, the results motivate continued experimentation into alternative surrogates to the uniform. In particular, *the results indicate that alternatives, such as the independence surrogate, may outperform the uniform surrogate (which is implicitly assumed by anomaly detection algorithms) in realistic classes of applications*.

6 Future Work

We currently are extending our detection system to incorporate several improvements and extensions. These include:

- Improving the θ-separation measures. This includes allowing application-dependent prior distributions on the processes to be factored into the computation explicitly.
- Replacing the single detector with a *k-tuple* of detectors ($k > 1$), as described in Section 2.5 and [9]. For example, one implementation of the k-tuple concept includes one detector for each of k measures, and a criterion for ordering the detectors. Then, if the first detector fails to distinguish a pair of transactions, rather than arbitrarily ordering the pair (as would the single detector), the next detector in the k-tuple is applied. The process continues until a detector in the k-tuple distinguishing the transaction pair is encountered.

- Extending the search algorithms supported. The exhaustive search employed in the experiments is not feasible in general (see section 4.3), motivating the development of algorithms that search heuristically the space of detectors. We currently are experimenting with several search algorithms, including: **greedy search** [10, 13], in which one subset element is added at each search stage based on local optimization; **simulated annealing** [11, 13], which occasionally selects weaker subsets in order to avoid local optima; and, **k-neighborhood search** [13], which imposes on the space a generalized neighborhood structure.

- Generalizing the equivalence classing transforms supported, to include value clustering and possibly other transformations.

Future testing will further validate the methodology. For example, a far more extensive suite of (N, P) model pairs will be tested. We wish to consider such issues as how similar can N and P be and still be distinguished by our detectors, and how do our θ-separation measures perform across this suite.

The techniques reported in this paper apply to many important application domains, including: computer security, computer networks, telecommunications, radio astronomy, and biomedical testing. We wish ultimately to use the system to analyze actual data from such problem domains.

Acknowledgements: We wish to thank the referees and editor for many useful suggestions which greatly improved the quality of this work. We wish to thank also Linda Adkins for her assistance in preparing the appendices.

References

[1] Anderson, J. P. *Computer Security Threat Monitoring and Surveillance*, James P. Anderson Co., Fort Washington, PA, April 1980.

[2] Baim, P. W., "A method for attribute selection in inductive learning systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, No. 6, pp. 888-896, 1988.

[3] Duda, R. and Hart, P., *Pattern Classification and Scene Analysis*, John Wiley and Sons, New York, 1973.

[4] Fukunaga, K., *Introduction to Statistical Pattern Recognition*, Academic Press, 1972.

[5] Garey, M. and Johnson, D., *Computers and Intractability*, W. H. Freeman, San Fransisco, 1979.

[6] Gonzalez, R.C. and Thomason, M.G., *Syntactic Pattern Recognition, an Introduction*, Addison-Wesley, Reading, Mass., 1978.

[7] Good, I. J., “The population frequencies of species and the estimation of population parameters”, *Biometrika*, vol. 40, parts 3 and 4, 1953, 237-264.

[8] Helman, P. and Liepins, G., “Statistical foundations of audit trail analysis for the detection of computer misuse”, *IEEE Transactions on Software Engineering*, vol. 19, no. 9, Sept. 1993, 886-901.

[9] Helman, P., “Prioritizing information for the discovery of phenomena”, *Technical Report CS94-8*, University of New Mexico.

[10] Horowitz, E. and Sahni, S., *Fundamentals of Computer Algorithms*, Computer Science Press, Potomac, MD, 1978.

[11] Johnson, D., Aragon, C., McGeoch, L., and Schevon, C., “Optimization by simulated annealing: An experimental evaluation; Part I, graph partitioning,” *Operations Research*, vol. 37, no. 6, November-December 1989, pp 865-892.

[12] Littlestone, N., “Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm,” *Machine Learning*, vol. 2, no. 2, pp. 285-318, 1988.

[13] Moret, B. and Shapiro, H., *Algorithms from P to NP*, Benjamin/Cummings Publishing Co., Redwood City, CA, 1991.

[14] Tou, J.T. and Gonzalez, R.C., *Pattern Recognition Principles*, Addison-Wesley, Reading, Mass., 1974.

[15] Watanabe, S., *Pattern Recognition: Human and Mechanical*, John Wiley and Sons, New York, 1985.

Appendix 1: Summary of Terminology and Notation

S – The space of all (potential) transactions.

N – The mundane process, with image in S.

P – The phenomenon process, with image in S.

DB — The training sample, generated by N. This is a sequence of transactions from S.

RS — The data to be analyzed, called the **rank stream**, generated by a blend of N and P. This is a sequence of transactions from S.

λ — The frequency with which N generates a transaction for RS. $(1 - \lambda)$ is the frequency with which P generates a transaction for RS.

θ_N – The distribution of process N. For $x \in S$, $\theta_N(x)$, also denoted $n(x)$ when θ_N is understood, is the probability $Pr\{x|N\}$ x is generated when N is operative.

θ_P – The distribution of process P. For $x \in S$, $\theta_P(x)$, also denoted $m(x)$ when θ_P is understood, is the probability $Pr\{x|P\}$ x is generated when P is operative.

θ – A distribution pair (θ_N, θ_P). Θ denotes the space of all such θ.

g – A prior density or probability function on Θ. G denotes the space of all such g.

$\sigma(S)$ – A prioritization (i.e., permutation) of S.

pd – A phenomenon detector, which maps S to $\Re$ (the real numbers).

$Class\ I\ detector$ – A pd which may not depend on the rank stream encountered.

$Class\ II\ detector$ – A pd which may depend on the rank stream encountered.

rsk – The Bayes risk associated with a pd or $\sigma(S)$.

Π – A subset of features.

$[x]_\Pi$ – The set of transactions identical to x on the features in Π.

$count(DB, x)$ – The number of occurrences in DB of x.

$n_f(x)$ – $count(DB, x)/|DB|$.

$count(DB, [x]_\Pi)$ – The number of occurrences in DB of members of $[x]_\Pi$.

$n_\Pi(x)$ – $count(DB, [x]_\Pi)/|DB|$.

$m_U(x)$ – The probability associated with x under the uniform surrogate.

$m_I(x)$ – The probability associated with x under the independence surrogate.

H_Π – The hypothesis that only features in Π are relevant to distinguishing processes N and P.

$\theta_{N,\Pi}$ – The distribution consistent with H_Π which maximizes $Pr\{DB|\theta_N\}$.

$\theta_{P,\Pi}$ – The distribution of the chosen surrogate, consistent with H_Π.

Appendix 2: Technical Details

I. Detector Classes

Following is formal definitions of the two detector classes introduced in Section **??**

More formally, we have the following definitions. θ_N and θ_P denote distribution vectors for the mundane and phenomenon processes, respectively, i.e., each assigns a probability to each $x \in S$. θ denotes a distribution pair (θ_N, θ_P), and Θ denotes the set of all such distribution pairs. G denotes the set $G = \{g | g$ is a (prior) density or probability function over $\Theta\}$.

A $Class\ I$ *detector* is a function $pd : G \times \{samples\ DB\} \times S \rightarrow \Re$.

Hence, such a pd's evaluation of each $x \in S$ does not depend on the RS which is encountered. Effectively, pd defines a single permutation $pd_{g,DB}(S) = \sigma(S)$ of S and pd's ranking of any RS is the restriction of $\sigma(S)$ to the distinct transactions appearing in RS. Practically, such a pd is instantiated to a map from $S \rightarrow \Re$ once DB is presented during the (off-line) training phase.

A $Class\ II$ *detector* is a function $pd : G \times \{samplesDB\} \times \{rank\ streams\ RS\} \times S \rightarrow \Re$.

Hence, such a pd's evaluation of $x \in S$ may depend on the RS which is encountered. Effectively, pd defines a permutation $pd_{g,DB}(RS) = \sigma(RS)$ for each RS encountered and these permutations may be independent, ie., $\sigma(RS)$ and $\sigma(RS')$ need not be consistent. Practically, such a pd is not fully instantiated to a map from $S \rightarrow \Re$ during the (off-line) training phase; analysis of RS must be performed in the course of on-line operation.

Since the prior $g \in G$ is fixed for an entire application, we typically suppress g as a detector argument and as an argument of the loss function defined below.

II. Formulae for Loss Function L

Following is the formal definition the loss function L introduced in Section 2

Define, for any transaction pair $\{x, y\}$,

$l(< x, y >; \theta) = Pr\{P|x; \theta\}Pr\{N|y; \theta\}$
$l(< x = y >; \theta) = 1/2(Pr\{P|x; \theta\}Pr\{N|y; \theta\} + Pr\{P|y; \theta\}Pr\{N|x; \theta\})$.

Then the loss $L_{x,y}$ incurred by pd on transaction pair $\{x, y\} \subset RS$ when the sample is DB, rank stream is RS, and the processes are characterized by θ is:

$$L_{x,y}(pd, DB, RS; \theta) = \begin{cases} l(< x, y >; \theta) & \text{if } pd(DB, RS, x) < pd(DB, RS, y) \\ l(< x = y >; \theta) & \text{if } pd(DB, RS, x) = pd(DB, RS, y) \\ l(< y, x >; \theta) & \text{if } pd(DB, RS, x) > pd(DB, RS, y) \end{cases}$$

Note that when pd denotes a $Class\ I$ detector, it is assumed constant in the argument RS. $L_{x,y}(pd, DB, RS; \theta)$ specifies that the loss incurred by a pd such that $pd(DB, RS, x) < pd(DB, RS, y)$ is the probability that, when θ characterizes the operative processes, x is generated by the phenomenon process while y is generated by the mundane process. This is an appropriate penalty, since it is exactly the probability that, when θ is operative, pd's ordering of x and y is a misdirection. (When $pd(x) = pd(y)$, the penalty is the average of the two directional penalties, reflecting the fact that in this case pd provides no guidance and x and y must be ranked by arbitrary means [e.g., a coin flip].)

Loss function L is defined as:

$$L(pd, DB, RS; \theta) = [1/(|RS|(|RS| - 1))] \sum_{\{x,y\} \subset RS} L_{x,y}(pd, DB, RS; \theta)$$

Notice that $|RS| * (|RS| - 1)$ is proportional to the maximum number of terms contributed by a rank stream of size $|RS|$; normalizing by the reciprocal of this quantity allows meaningful comparisons of loss incurred on rank streams of different sizes.

III. Proofs of Theorem and its Corollaries

Following is the proofs of the Theorem and its corollaries which appear in Section 3.

Theorem: CSMP is NP-complete.

Proof: CSMP is in the class NP, since we can in polynomial time produce a subset Π of features and calculate its pairwise count spread. To establish CSMP NP-complete, we reduce to it a restricted version of SRP. The proof of Theorem 3 in [8] demonstrates the following version of the singleton reduction problem to be NP-complete.

INSTANCE: Database DB of D transactions, each consisting of D features such that every feature value is either 1 or some unique integer value appearing nowhere else in DB, and integers K and s, each less than or equal to D.

QUESTION: Is there a subset Π consting of K or more features such that the number of singleton transactions, when DB is projected onto Π, is s or fewer.

We view a database of T transactions as a $T \times D$ matrix, with row i specifying the D feature values for transaction t_i. The terms transaction and matrix row will be used interchangeably. Our constructed instance of CSMP consists of transactions over D integer-valued features and hence corresponds to a D-column integer matrix. The constructed matrix is described in terms of two main matrix components, C_{enf}, which enforces the restriction that only cardinality K subsets can attain the target spread value B (specified later), and C_{sing}, which ensures that, among cardinality K subsets, only ones which yield s or fewer singletons against DB can attain spread B.

We describe first the matrix component C_{sing}. The rows of the $D \times D$ matrix DB are replicated so for each row $< a_{i1}, a_{i2}, \ldots, a_{iD} >$ in DB there are $D+1$ identical rows in C_{sing}. Hence, if a transaction has count q in $DB[\Pi]$ (that is, DB projected onto Π) it has count $q(D+1)$ in $C_{sing}[\Pi]$. An additional collection of $D-1$ rows is constructed, such that each entry in these rows is an integer appearing no where else in the constructed instance. This collection is replicated $D+1$ times and added to the matrix component C_{sing}.

Let Π be any subset of features and let g be the number of singletons in $DB[\Pi]$. Then the counts in $C_{sing}[\Pi]$ are as follows:

$D-1+g$ classes with count $D+1$. These correspond to the g singletons in $DB[\Pi]$ plus the $D-1$ unique-valued rows replicated so each has count $D+1$.

1 class with count $(D+1)(D-g)$. This corresponds to the collection of transactions in $DB[\Pi]$ which are not singletons. Note that since all entries of the nonsingletons rows projected onto Π must be 1 (else the transactions would be singletons), they collapse to a single class.

Hence, for any Π such that $DB[\Pi]$ yields g singletons, the spread of the counts of the Π-classes within C_{sing} is given by $(D-1+g)[(D+1)((D-g)-1)]$, which represents the distance between the counts of the $(D-1+g)$ pairs of classes with unequal counts. The derivative of this expression with respect to g is $(D+1)[D-(D-1)-2g-1] = (D+1)(-2g)$. It therefore follows immediately that the the spread of the counts of classes within C_{sing} is strictly increasing as the number g of singletons decreases (for $g \geq 0$). Note further that $U_{sing} = (D-1)^2(D+1)$ is a (not necessarily achievable) upper bound on the spread of count pairs within C_{sing}.

We next specify the matrix component C_{enf} which ensures that only Π of cardinality K can achieve the target spread B or greater. We first build a subcomponent of C_{enf} as follows.

1. Construct a $D \times D$ matrix such that the off-diagonal entries are all set to some (single) integer value which appears no where else in the construction. The on-diagonal entries are distinct from each other, as well as from all other values appearing in the construction. Observe that when this subcomponent of C_{enf} is projected onto any set Π of cardinality c, there are c singletons (if $A_j \in \Pi$ then the ith row of this subcomponent is a singleton) and the remaining $D-c$ rows collapse to a single class with count $D-c$.

2. Add $2(D-K)$ rows consisting of all unique integers appearing no where else in the construction. When these rows are projected onto any Π, $2(D-K)$ singletons result.

3. Add $D+1$ rows with all entries set to a single integer appearing no where else in the construction. When these rows are projected onto any Π, a single class with count $D+1$ results.

The construction of component C_{enf} is completed by replicating the structure (but not re-using the same values between copies) of subcomponent formed from the 3 steps above, so there are a total of $J = U_{sing} + (D^2 + D)(6D^2 + 2)$ of these subcomponents in C_{enf}.

It follows from the preceding observations that when matrix component C_{enf} is projected onto any set Π of cardinality c, there are:

1. $J(c + 2(D - K))$ class counts of 1.
2. J class counts of $D - c$.
3. J class counts of $D + 1$.

Hence, the spread of the class counts for pairs within C_{enf} is

$$\begin{aligned} & J^2[((D+1) - (D-c)) + (c + 2(D-K))(D + ((D-c) - 1))] \\ = {} & J^2[2D^2 - c^2 - 4KD + 2cK + 2K - 2D] \end{aligned}$$

Differentiating with respect to c we obtain $J^2[-2c + 2K]$. It then follows immediately that the spread of the counts of classes within C_{enf} is maximized when we project onto any Π of cardinality $c = K$ and is strictly less than this maximum value for any Π' of cardinality unequal to K.

Since counts are integral it follows also that the difference of the spread achieved on the count pairs within C_{enf} by any cardinality K set Π and that achieved by any Π' of cardinality other than K is at least J^2. We have already observed that U_{sing} is an upper bound on the total spread that can be achieved by pairs of counts contained in C_{sing}. Note that $(D^2 + D)$ bounds the largest class count that can be achieved in the constructed instance and that $(6D^2 + 2)$ bounds the number of pairs of counts that can be formed by selecting one count from C_{sing} and one count from any one fixed subcomponent (i.e., any one of the J copies) of C_{enf}. Consequently, $J(D^2 + D)(6D^2 + 2)$ is a (gross) upper bound on the total spread that can be achieved by the set of all pairs with one count in C_{sing} and one count in C_{enf}. By the way J was constructed, $J^2 > U_{sing} + J(D^2 + D)(6D^2 + 2)$. Consequently, if Π' is of cardinality other than K it cannot achieve enough total spread on pairs not totally contained in C_{enf} to make up for the at least J^2 advantage on those pairs totally contained in C_{enf} held by any Π of cardinality K. Consequently, every Π of cardinality K achieves a higher total count spread than every Π' not of cardinality K.

We now claim that Π maximizes total spread if and only if, among cardinality K sets, it minimizes the number of singletons. We already have observed that the spread of counts contained in C_{sing} strictly increases as the number of singletons decreases. Further, the exact identity of the counts contained in C_{enf} (and hence their spread) is identical for all cardinality K Π. Hence, the only remaining consideration is the spread of counts between the two components. Since the largest class count in C_{enf} always is $D+1$ and since C_{sing} consists of $(D-1+g)$ classes of count $D+1$ and 1 class of count $(D+1)(D-g)$, the inter-component spread also strictly increases as the number of singletons g in $DB[\Pi]$ decreases, as Π varies over the cardinality K subsets of features. To see this, observe that since every count m appearing in C_{enf} is at most $D+1$, the contribution of this count to the inter-component total when Π yields $g < D$ singletons is

$$((D+1)(D-g)-m)+(D-1+g)((D+1)-m) = ((D+1)(2D-1)-(D+g)m)$$

The contribution is $(2D-1)((D+1)-m)$ when $g = D$. It is immediately apparent that this contribution is strictly increasing as g decreases from $D-1$ to 0. While the contributions are identical when $g = D$ and $g = D-1$, note that a count of $g = D-1$ singletons in a database of D transactions is impossible (if $D-1$ transactions are unique, so too must be the last transaction, yielding D singletons), and hence the claim is established.

To complete the construction, we set the target spread value B to $Sum_{sing} + Sum_{enf} + Sum_{inter}$, where

$$Sum_{sing} = (D-1+s)[(D+1)((D-s)-1)]$$

which represents the spread of counts contained in C_{sing} when a Π yielding s singletons in $DB[\Pi]$ is selected,

$$Sum_{enf} = J^2[2D^2+K^2-4KD+2K-2D]$$

which represents the spread of counts contained in C_{enf} when a Π of cardinality K is selected (J is as specified in the construction of C_{enf}), and

$$Sum_{inter} = J(K + 2(D - K))[(D - 1 + s)D + ((D + 1)(D - s) - 1)]$$
$$+ J[(D - 1 + s)(K + 1) + ((D + 1)(D - s) - (D - K)) + (D + 1)(D - s - 1)]$$

which represents the spread of inter-component count pairs when a Π of cardinality K, yielding s singletons in $DB[\Pi]$, is selected.

Hence, spread B is achieved by any cardinality K subset Π that yields exactly s singletons in the source instance. It follows from the spread's strict monotonicity in the number g of singletons, and the construction of component C_{enf}, that Π achieves B or greater spread if and only if Π is of cardinality K and yields s or fewer singletons in the target SRP instance. QED

Corollary 1: CSMP with an explicit constraint on the subset cardinality is NP-complete.

Proof: Transformation from SRP to constrained CSMP is accomplished by constructing C_{sing} as in the previous proof, setting CSMP's cardinality constraint to the constraint K of the SRP target instance, and setting B to Sum_{sing}. QED

Corollary 2: CSMP is NP-complete if class counts are divided by class cardinality, as in measure M_2. This result holds for both the cardinality-constrained and unconstrained versions of the problem.

Proof: For the constrained version, no modification of the construction specified in Corollary 1's proof is necessary (except scaling of B), since feature domains can be assumed of equal cardinality, implying that class cardinality is identical for all cardinality K subsets and hence has only a scaling effect on the calculations. Only minor modifications to the construction specified in the theorem's proof are required for the constrained version. For example, the values and replication factor in C_{enf} must be modified to ensure that a cardinality K subset is chosen. QED

Appendix 3: Process Distributions

This appendix presents the marginal distributions for the processes N and P used in each experiment. Only the distributions of the independent features are given; the body of the paper describes how the dependent features are derived from the independent ones.

Generic Models, Experiment 1

A. 5 fields, domain size = 10

Process N:

For $i = 1, 2$
$Pr\{A_i = v\} = 0.1111$, for $v = 1, 2, 3$
$Pr\{A_i = v\} = 0.1665$, for $v = 4, 5$
$Pr\{A_i = v\} = 0.0833$, for $v = 6, 7, 8, 9, 10$

For $i = 3, 4, 5$
$Pr\{A_i = v\} = 0.1$, for $v = 1, 2, ..., 10$

Process P:

For $i = 1, 2, ..., 5$
$Pr\{A_i = v\} = 0.1$, for $v = 1, 2, ..., 10$

B. 5 fields, domain size = 4

Process N:

For $i = 1, 2$
$Pr\{A_i = v\} = 0.5000$, for $v = 1$
$Pr\{A_i = v\} = 0.1667$, for $v = 2, 3, 4$

For $i = 3, 4, 5$

$Pr\{A_i = v\} = 0.25$, for $v = 1, 2, 3, 4$

Process P:

For $i = 1, 2, 3, 4, 5$

$Pr\{A_i = v\} = 0.25$, for $v = 1, 2, 3, 4$

C. 3 fields, domain size = 10

Distributions are the same as in Experiment 1.A, restricted to features A_1, A_2, A_3

D. 3 fields, domain size = 4

Process N:

Distributions are the same as in Experiment 1.B, restricted to features A_1, A_2, A_3

Generic Models, Experiment 2

A. 5 fields, domain size = 10

Processes N and P:

For $i = 1, 3, 4, 5$

$Pr\{A_i = 1\} = 0.001953$

$Pr\{A_i = 2\} = 0.017578$

$Pr\{A_i = 3\} = 0.070313$

$Pr\{A_i = 4\} = 0.164062$

$Pr\{A_i = 5\} = 0.246094$

$Pr\{A_i = 6\} = 0.246094$

$Pr\{A_i = 7\} = 0.164062$

$Pr\{A_i = 8\} = 0.070313$

$Pr\{A_i = 9\} = 0.017578$

$Pr\{A_i = 10\} = 0.001953$

A_2 dependent on A_1; dependency differs in N and P, as described in body of the paper.

B. 5 fields, domain size = 4

Processes N and P:

For $i = 1, 3, 4, 5$
$Pr\{A_i = 1\} = 0.125$
$Pr\{A_i = 2\} = 0.375$
$Pr\{A_i = 3\} = 0.375$
$Pr\{A_i = 4\} = 0.125$

A_2 dependent on A_1; dependency differs in N and P, as described in body of the paper.

C. 3 fields, domain size = 10

Distributions are the same as in Experiment 2.A, restricted to features A_1, A_2, A_3

D. 3 fields, domain size = 4

Distributions are the same as in Experiment 2.B, restricted to features A_1, A_2, A_3

Computer Intrusion Models, Experiment 1

Process N:

$Pr\{User = v\} = 0.1225$, for $v = 1, 2, 3, 4$
$Pr\{User = v\} = 0.02$, for $v = 5$
$Pr\{User = v\} = 0.098$, for $v = 6, 7, ..., 10$

$Pr\{Time\ of\ Day = v\} = 0.01$, for $v = 1, 2, ..., 100$

$Pr\{Terminal = v\} = 0.05$, for $v = 1, 2, ..., 20$

$Pr\{System\ Load = v\} = 0.01$, for $v = 1, 2, ..., 100$

$Command$ dependent on $User$ as described in body of the paper.

Process P:

$Pr\{User = v\} = 0.0833$, for $v = 1, 2, 3, 4$

$Pr\{User = v\} = 0.3333$, for $v = 5$

$Pr\{User = v\} = 0.0665$, for $v = 6, 7, ..., 10$

Remaining independent fields distributed as in N; $Command$ dependent on $User$ as described in body of the paper.

Computer Intrusion Models, Experiment 2

Processes N and P:

$Pr\{User = 1\} = 0.001953$

$Pr\{User = 2\} = 0.017578$

$Pr\{User = 3\} = 0.070313$

$Pr\{User = 4\} = 0.164062$

$Pr\{User = 5\} = 0.246094$

$Pr\{User = 6\} = 0.246094$

$Pr\{User = 7\} = 0.164062$

$Pr\{User = 8\} = 0.070313$

$Pr\{User = 9\} = 0.017578$

$Pr\{User = 10\} = 0.001953$

$Pr\{\, Time\ of\ Day = v\} = 0.0001953$, for $v = 1, 2, ..., 10$

$Pr\{\, Time\ of\ Day = v\} = 0.0017578$, for $v = 11, 12, ..., 20$

$Pr\{\, Time\ of\ Day = v\} = 0.0070313$, for $v = 21, 22, ..., 30$

$Pr\{\, Time\ of\ Day = v\} = 0.0164062$, for $v = 31, 32, ..., 40$

$Pr\{\, Time\ of\ Day = v\} = 0.0246094$, for $v = 41, 42, ..., 50$

$Pr\{\, Time\ of\ Day = v\} = 0.0246094$, for $v = 51, 52, ..., 60$

$Pr\{\, Time\ of\ Day = v\} = 0.0164062$, for $v = 61, 62, ..., 70$

$Pr\{\, Time\ of\ Day = v\} = 0.0070313$, for $v = 71, 72, ..., 80$

$Pr\{\, Time\ of\ Day = v\} = 0.0017578$, for $v = 81, 82, ..., 90$

$Pr\{\, Time\ of\ Day = v\} = 0.0001953$, for $v = 91, 92, ..., 100$

$Pr\{Terminal = v\} = 0.0009765$, for $v = 1, 2$

$Pr\{Terminal = v\} = 0.0008789$, for $v = 3, 4$

$Pr\{Terminal = v\} = 0.0351565$, for $v = 5, 6$

$Pr\{Terminal = v\} = 0.0820310$, for $v = 7, 8$

$Pr\{Terminal = v\} = 0.1204700$, for $v = 9, 10$

$Pr\{Terminal = v\} = 0.1204700$, for $v = 11, 12$

$Pr\{Terminal = v\} = 0.0820310$, for $v = 13, 14$

$Pr\{Terminal = v\} = 0.0351565$, for $v = 15, 16$

$Pr\{Terminal = v\} = 0.0008789$, for $v = 17, 18$

$Pr\{Terminal = v\} = 0.0009765$, for $v = 19, 20$

$Pr\{\, System\ Load = v\} = 0.0001953$, for $v = 1, 2, ..., 10$

$Pr\{\, System\ Load = v\} = 0.0017578$, for $v = 11, 12, ..., 20$

$Pr\{\, System\ Load = v\} = 0.0070313$, for $v = 21, 22, ..., 30$

$Pr\{\, System\ Load = v\} = 0.0164062$, for $v = 31, 32, ..., 40$

$Pr\{\, System\ Load = v\} = 0.0246094$, for $v = 41, 42, ..., 50$

$Pr\{\ System\ Load = v\} =$ 0.0246094, for $v = 51, 52, ..., 60$

$Pr\{\ System\ Load = v\} =$ 0.0164062, for $v = 61, 62, ..., 70$

$Pr\{\ System\ Load = v\} =$ 0.0070313, for $v = 71, 72, ..., 80$

$Pr\{\ System\ Load = v\} =$ 0.0017578, for $v = 81, 82, ..., 90$

$Pr\{\ System\ Load = v\} =$ 0.0001953, for $v = 91, 92, ..., 100$

$Command$ dependent on $User$; dependency differs in N and P, as described in body of the paper.

Computer Intrusion Models, Experiment 3

Process N:

Features $User$, $Time\ of\ Day$, $Terminal$, $System\ Load$ distributed as in Process N of Experiment 2. Feature $Command$ here is an independent feature and is distributed as follows.

$Pr\{Command = v\} =$ 0.000651, for $v = 1, 2, 3$

$Pr\{Command = v\} =$ 0.005893, for $v = 4, 5, 6$

$Pr\{Command = v\} =$ 0.023437, for $v = 7, 8, 9$

$Pr\{Command = v\} =$ 0.041015, for $v = 10, 11, 12, 13$

$Pr\{Command = v\} =$ 0.082031, for $v = 14, 15, 16$

$Pr\{Command = v\} =$ 0.082031, for $v = 17, 18, 19$

$Pr\{Command = v\} =$ 0.041015, for $v = 20, 21, 22, 23$

$Pr\{Command = v\} =$ 0.023437, for $v = 24, 25, 26$

$Pr\{Command = v\} =$ 0.005893, for $v = 27, 28, 29$

$Pr\{Command = v\} =$ 0.000488, for $v = 30, 31, 32, 33$

Process P:

Features $User$, $Time\ of\ Day$, $Terminal$, $System\ Load$ distributed as in Process P of Experiment 2. Feature $Command$ is dependent on $User$ as described in body of the paper.

Appendix 4: Experimental Results

This appendix presents selected numerical results from the experiments. The format of the results is the same for each experiment:

The first table reports the subset preferred by each version of the θ-separation measure (i.e., measure with uniform surrogate computed over all transaction pairs in the training data, measure with independence surrogate computed over all transaction pairs in the training data, measure with uniform surrogate computed over a random sample of transaction pairs in the training data, measure with independence surrogate computed over a random sample of transaction pairs in the training data.)

The second table reports the observed prioritization, averaged over 100 rank streams, for each of the corresponding detectors, along with err(optimal) (the error incurred by the theoretically optimal detector), err(naive) (the error incurred by the naive detector), and err(best subset) (the error incurred by the best subset detector).

The third table reports the number of rank streams (out of 100) for which the indicated subset (paired with each of the surrogates) incurred a smaller prioritization error than did the naive detector.