Data Mining for Analysis of Rare Events: A Case Study in Security, Financial and Medical Applications

Aleksandar Lazarević,
Jaideep Srivastava, Vipin Kumar

Army High Performance Computing Research Center
Department of Computer Science
University of Minnesota

PAKDD-2004 Tutorial

AHPCRC

Introduction

- We are drowning in the deluge of data that are being collected world-wide, while starving for knowledge at the same time*
- Despite the enormous amount of data, particular events of interest are still quite rare
- Rare events are events that occur very infrequently, i.e. their frequency ranges from 0.1% to less than 10%
- However, when they do occur, their consequences can be quite dramatic and quite often in a negative sense

"Mining needle in a haystack. So much hay and so little time"

* - J. Naisbitt, Megatrends: Ten New Directions Transforming Our Lives. New York: Warner Books, 1982.

Related problems

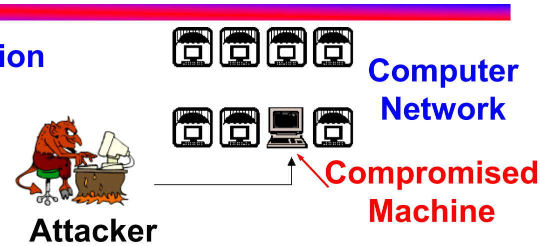
- Chance discovery
  - "chance is defined as some event which is significant for decision-making in a specified domain" Yukio Ohsawa
  - chance event may be either positive (opportunity), or negative (potential risk)
  - Chance Discovery – learning, explaining and discovering such chance events, typically rare to find
- Novelty Detection
- Exception Mining
- Black Swan* [Taleb'04]

AHPCRC

* N. Talleb, The Black Swan: Why Don't We Learn that We Don't Learn?, draft 2004

Applications of Rare Classes

- Network intrusion detection
  - number of intrusions on the network is typically a very small fraction of the total network traffic



- Credit card fraud detection
  - Millions of regular transactions are stored, while only a very small percentage corresponds to fraud

- Medical diagnostics
  - When classifying the pixels in mammogram images, cancerous pixels represent only a very small fraction of the entire image

AHPCRC

Industrial Applications of Rare Classes

- Insurance Risk Modeling (e.g. Pednault, Rosen, Apte '00)
  - Claims are rare but very costly
- Web mining
  - Less than 3% of all people visiting Amazon.com make a purchase
- Targeted Marketing (e.g. Zadrozny, Elkan '01)
  - Response is typically rare but can be profitable
- Churn Analysis (e.g. Mamitsuka and Abe '00)
  - Churn is typically rare but quite costly
- Hardware Fault Detection (e.g. Apte, Weiss, Grout 93)
  - Faults are rare but very costly
- Airline No-show Prediction (e.g. Lawrence, Hong, et al '03)
  - Disease is typically rare but can be deadly

AHPCRC

Limitations of Standard Data Mining Schemes

- Standard approaches for feature selection and construction do not work well for rare class analysis
- While most normal events are similar to each other, rare events are quite different from one another
  - regular credit transaction are fairly standard, while fraudulent ones vary from the standard ones in many different ways
- Metrics used to evaluate normal event detection methods
  - Accuracy is not appropriate for evaluating methods for rare event detection
- In many applications data keeps arriving in an ongoing stream, and there is a need to detect rare events on the fly, with models built only on the events seen so far

AHPCRC

Evaluation of Rare Class Problems – F-value

- Accuracy is not sufficient metric for evaluation
  - Example: network traffic data set with 99.9% of normal data and 0.1% of intrusions
  - Trivial classifier that labels everything with the normal class can achieve 99.9% accuracy !!!!!

Confusion matrix		Predicted class	
		NC	C
Actual class	NC	TN	FP
	C	FN	TP

rare class – C
normal class – NC

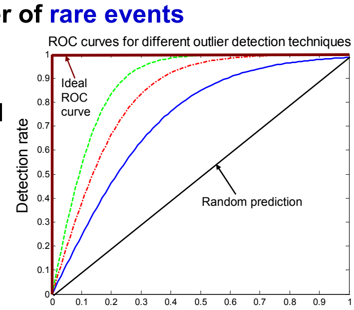
- Focus on both recall and precision
  - Recall (R) = TP/(TP + FN)
  - Precision (P) = TP/(TP + FP)
- F – measure = 2*R*P/(R+P)

Evaluation of Rare Class Problems - ROC

Confusion matrix		Predicted class	
		NC	C
Actual class	NC	TN	FP
	C	FN	TP

rare class – C
normal class – NC

- Standard measures for evaluating rare class problems:
  - *Detection rate (Recall)* - ratio between the number of correctly detected rare events and the total number of rare events
  - *False alarm (false positive) rate* – ratio between the number of data records from majority class that are misclassified as rare events and the total number of data records from majority class
  - *ROC Curve* is a trade-off between detection rate and false alarm rate



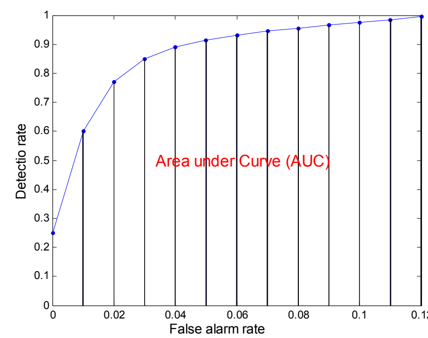
AHPCRC

Evaluation of Rare Class Problems - AUC

Confusion matrix		Predicted class	
		NC	C
Actual class	NC	TN	FP
	C	FN	TP

rare class – C
normal class – NC

Area under the ROC curve (AUC) is computed using a form of the trapezoid rule.



Equivalent Mann-Whitney two-sample statistics:

$$\hat{A} = \frac{1}{m \cdot n} \sum_{i=1}^{m} \sum_{j=1}^{n} I(r_i^-, r_j^+), \quad I(r^-, r^+) = \begin{cases} 1 & \text{if } r^- > r^+ \\ \frac{1}{2} & \text{if } r^- = r^+ \\ 0 & \text{if } r^- < r^+ \end{cases}$$

AHPCRC

m ratings of negative cases r^-
n ratings of positive cases r^+

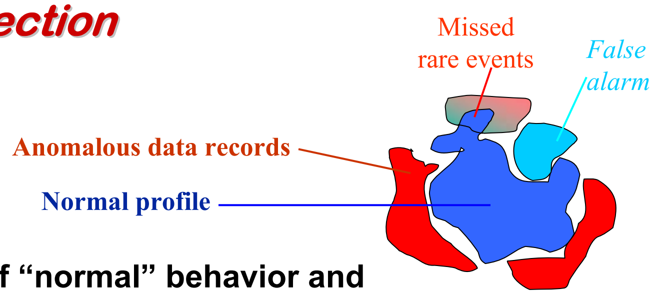
Example: in naïve Bayes, rating may be the posterior probability of the positive class

Major Techniques for Detecting Rare Events

- **Unsupervised techniques**
  - Deviation detection, outlier analysis, anomaly detection, exception mining
  - Analyze each event to determine how similar (or dissimilar) it is to the majority, and their success depends on the choice of similarity measures, dimension weighting
- **Supervised techniques**
  - Mining rare classes
  - Build a model for rare events based on labeled data (the training set), and use it to classify each event
  - Advantage: they produce models that can be easily understood
  - Drawback: The data has to be labeled
- **Other techniques – association rules, clustering**

AHPCRC

Unsupervised Techniques – Anomaly Detection

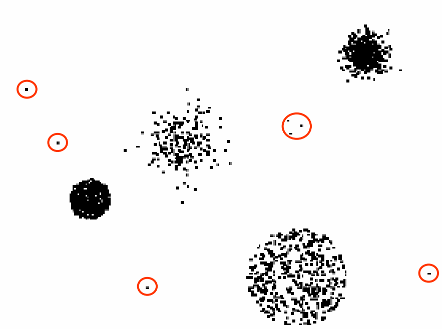


- Build models of "normal" behavior and detect anomalies as deviations from it
- Possible high false alarm rate - previously unseen (yet legitimate) data records may be recognized as anomalies
- Two types of techniques
  - with access to normal data
  - with NO access to normal data (not known what is "normal")

AHPCRC

Outlier Detection Schemes

- Outlier is defined as a data point which is very different from the rest of the data based on some measure
- Detect novel attacks/intrusions by identifying them as deviations from "normal" behavior
  - Identify normal behavior
  - Construct useful set of features
  - Define similarity function
  - Use outlier detection algorithm
    - Statistics based approaches
    - Distance based approaches
      - Nearest neighbor approaches
      - Clustering based approaches
      - Density based schemes
    - Model based schemes



AHPCRC

Statistics Based Outlier Detection Schemes

- Statistics based approaches – data points are modeled using stochastic distribution ⇒ points are determined to be outliers depending on their relationship with this model
  - With high dimensions, difficult to estimate distributions
- Major approaches
  - Finite Mixtures
  - BACON
  - Using probability distribution
  - Information Theory measures

AHPCRC

Statistics Based Outlier Detection Schemes

- Using Finite Mixtures – SmartSifter (SS)*
- SS uses a probabilistic model as a representation of underlying mechanism of data generation.
  - Histogram density used to represent a probability density for categorical attributes
    - SDLE (Sequentially Discounting Laplace Estimation) for learning histogram density for categorical domain
  - Finite mixture model used to represent a probability density for continuous attributes
    - SDEM (Sequentially Discounting Expectatioan and Maximizing) for learning finite mixture for continuous domain
- SS gives a score to each example x_i (input into the model) on the basis of the learned model, measuring how large the model has changed after the learning

AHPCRC

* K. Yamanishi, On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms, KDD 2000

Statistics Based Outlier Detection Schemes

- BACON* Basic Steps:
  1. Select an initial basic subset of size *m* free of outliers
     - Initial subset selected based on *Mahalanobis* distances
     - Initial subset selected based on distances from the medians
  2. Fit the model to the subset and for $\forall x_i$ compute the discrepancies

     $$d_i(\mu_b, \Sigma_b) = \sqrt{(x_i - \mu_b)^T \cdot \Sigma_b^{-1} \cdot (x_i - \mu_b)}$$
  3. Set the new basic subset to all points with discrepancy less than $c_{npr}\ \chi^2_{p,\alpha}$ where $\chi^2_{p,\alpha}$ is $1 - \alpha$ percentile of the χ^2 distribution with *p* degrees of freedom, $c_{npr} = c_{np} + c_{hr}$ is a *correction factor,* $c_{hr} = \max\{0, (h-r)/(h+r)\}$; $h=[(n+p+1)/2]$; *r* is the size of the current basic subset $c_{np} = 1 + \frac{p+1}{n-p} + \frac{1}{n-h-p}$
  4. Iterate steps 2&3 until the the size of basic subset no longer changes
  5. List observations excluded by the final basic subset as outliers

AHPCRC

* N. Billor, A. Hadi, P. Velleman, BACON: blocked adaptive computationally efficient outlier nominators, *Computational Statistics & Data Analysis*, 34, 279-298, 2000.

Statistics Based Outlier Detection Schemes

- Using Probability Distributions*
- Basic Assumption: # of normal elements in the data is significantly larger then # of anomalies
- Distribution for the data *D* is given by:
  - $D = (1-\lambda)\cdot M + \lambda \cdot A$
    M - majority distribution, A - anomalous distribution
  - M_t, A_t sets of normal, anomalous elements respectively
  - Compute likelihood $L_t(D)$ of distribution *D* at time *t*
  - Measure how likely each element x_t is outlier:
    - $M_t = M_{t-1} \setminus \{x_t\}$, $A_t = A_{t-1} \cup \{x_t\}$
    - Measure the difference $(L_t - L_{t-1})$

AHPCRC

* E. Eskin, Anomaly Detection over Noisy Data using Learned Probability Distributions, ICML 2000

Statistics Based Outlier Detection Schemes

- *Using Information-Theoretic Measures**
- **Entropy measures the uncertainty (impurity) of data items**
  - The entropy is smaller when the class distribution is skewer
  - Each *unique* data record represents a class => the smaller the entropy the fewer the number of different records (higher redundancies)
  - If the entropy is large, data is partitioned into *more regular* subsets
  - Any deviation from achieved entropy indicates potential intrusion
  - Anomaly detector constructed on data with smaller entropy will be simpler and more accurate
- **Conditional entropy H(X|Y) tells how much uncertainty remains in sequence of events X after we have seen subsequence Y ($Y \in X$)**
- **Relative Conditional Entropy**

AHPCRC

* W. Lee, et al, Information-Theoretic Measures for Anomaly Detection, IEEE Symposium on Security 2001

Distance based Outlier Detection Schemes

- ***Nearest Neighbor (NN) approach*[1,2]**
  - **For each data point *d* compute the distance to the *k-th* nearest neighbor d_k**
  - **Sort all data points according to the distance d_k**
  - **Outliers are points that have the largest distance d_k and therefore are located in the more sparse neighborhoods**
  - **Usually data points that have top *n*% distance d_k are identified as outliers**
    - ***n* – user parameter**
  - **Not suitable for datasets that have modes with varying density**

AHPCRC

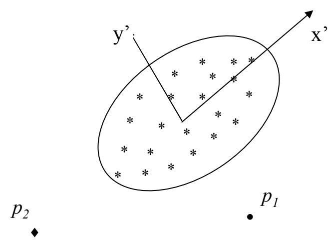
1. Knorr, Ng,Algorithms for Mining Distance-Based Outliers in Large Datasets, VLDB98
2. S. Ramaswamy, R. Rastogi, S. Kyuseok: Efficient Algorithms for Mining Outliers from Large Data Sets, ACM SIGMOD Conf. On Management of Data, 2000.

Distance based Outlier Detection Schemes

- ***Mahalanobis-distance based approach***
  - **Mahalanobis distance is more appropriate for computing distances with skewed distributions**

$$d_M = \sqrt{(p-\mu)^T \cdot \Sigma^{-1} \cdot (p-\mu)}$$

  - **Example:**
    - **In Euclidean space, data point p_1 is closer to the origin than data point p_2**
    - **When computing Mahalanobis distance, data points p_1 and p_2 are equally distant from the origin**



Density based Outlier Detection Schemes

- ***Local Outlier Factor (LOF) approach****
  - **For each data point O compute compute the distance to the k-th nearest neighbor (*k-distance*)**
  - **Compute *reachability distance* (*reach-dist*) for each data example O with respect to data example p as:**

reach-dist(O,p) = max{_k-distance(p)_, _d_(O,p)}

  - **Compute *local reachability density* (*lrd*) of data example O as inverse of the average reachabaility distance based on the *MinPts* nearest neighbors of data example O**

$$lrd(O) = \frac{MinPts}{\sum_p reach\_dist_{MinPts}(O,p)}$$

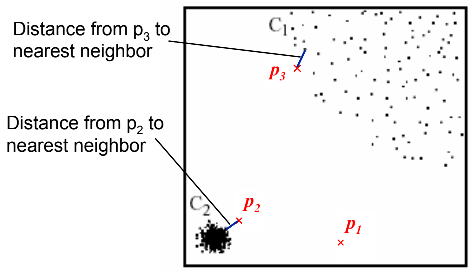
  - **Compute *LOF* of example *p* as the average of the ratios of the density of example *p* and the density of its nearest neighbors**

$$LOF(O) = \frac{1}{MinPts} \cdot \sum_p \frac{lrd(p)}{lrd(O)}$$

*- Breunig, et al, LOF: Identifying Density-Based Local Outliers, KDD 2000.

Advantages of Density based Schemes

- *Local Outlier Factor (LOF) approach*
  - Example:



In the *NN* approach, p_2 is not considered as outlier, while the *LOF* approach find both p_1 and p_2 as outliers

NN approach may consider p_3 as outlier, but LOF approach does not

Clustering based outlier detection schemes*

- Radius ω of proximity is specified
- Two points x_1 and x_2 are "near" if $d(x_1, x_2) \leq \omega$
- Define $N(x)$ – number of points that are within ω of x
- Time Complexity $O(n^2) \Rightarrow$ approximation of the algorithm
- Fixed-width clustering is first applied
  - The first point is a center of a cluster
  - If every subsequent point is "near" add to a cluster
    - Otherwise create a new cluster
  - Approximate $N(x)$ with $N(c)$
  - Time Complexity – $O(cn)$, c - # of clusters
- Points in small clusters - anomalies

* E. Eskin et al., A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data, 2002

Clustering based outlier detection schemes

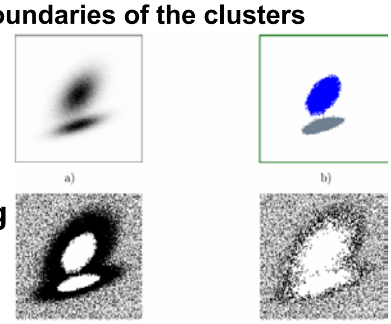
- K-nearest neighbor + canopy clustering approach *
- Compute the sum of distances to the k nearest neighbors (*k*-NN) of each point
  - Points in dense regions – small k-NN score
  - *k* has to exceed the frequency of any given attack type
  - Time complexity $O(n^2)$
- Speed up with *canopy clustering* that is used to split the entire space into small subsets (canopies) and then to check only the nearest points within the canopies
- Apply fixed width clustering and compute distances within clusters and to the centers of other clusters

AHPCRC

* E. Eskin et al., A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data, 2002

Clustering based outlier detection schemes

- FindOut algorithm* by-product of *WaveCluster*
- Main idea: Remove the clusters from original data and then identify the outliers
- Transform data into multidimensional signals using wavelet transformation
  - High frequency of the signals correspond to regions where is the rapid change of distribution – boundaries of the clusters
  - Low frequency parts correspond to the regions where the data is concentrated
- Remove these high and low frequency parts and all remaining points will be outliers



AHPCRC

* D. Yu, G. Sheikholeslami, A. Zhang, FindOut: Finding Outliers in Very Large Datasets, 1999.

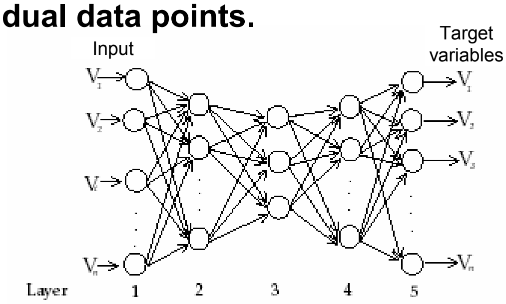
Model based outlier detection schemes

- Use a prediction model to learn the normal behavior
- Every deviation from learned prediction model can be treated as anomaly or potential intrusion
- Recent approaches:
  - Neural networks
  - Unsupervised Support Vector Machines (SVMs)

AHPCRC

Neural networks for outlier detection*

- Use a replicator 4-layer feed-forward neural network (RNN) with the same number of input and output nodes
- Input variables are the output variables so that RNN forms a compressed model of the data during training
- A measure of outlyingness is the reconstruction error of individual data points.

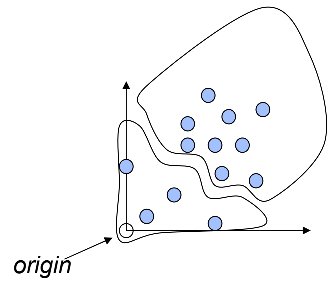


AHPCRC

* S. Hawkins, et al. Outlier detection using replicator neural networks, DaWaK02 2002.

Unsupervised Support Vector Machines for Outlier Detection

- Unsupervised SVMs attempt to separate the entire set of training data from the origin, i.e. to find a small region where most of the data lies and label data points in this region as one class
- Parameters
  - Expected number of outliers
  - Variance of rbf kernel
    - As the variance of the rbf kernel gets smaller, the number of support vectors is larger and the separating surface gets more complex



push the hyper plane away from origin as much as possible

* E. Eskin et al., A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data, 2002.

* A. Lazarevic, et al., A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection, SIAM 2003

AHPCRC

Supervised Classification for Rare Classes

- Standard classification models are not suitable for rare classes
- Models must be able to handle skewed class distributions
- Learning from data streams - sequences of events
- Key approaches:
  - *Manipulating data records (oversampling / undersampling / generating artificial examples)*
  - *Design of new algorithms (SHRINK, PN-rule, CREDOS)*
  - *Case specific feature/rule weighting*
  - *Boosting based algorithms (SMOTEBoost, RareBoost)*
  - *Cost sensitive classification (MetaCost, AdaCost, CSB, SSTBoost)*
  - *Emerging Patterns*
  - *Query/Active Learning Approach*
  - *Internally bias discrimination*
  - *Clustering based classification*

AHPCRC

Manipulating Data Records

- Over-sampling the rare class*
  - Make the duplicates of the rare events until the data set contains as many examples as the majority class => balance the classes
  - Does not increase information but increase misclassification cost
- Down-sizing (undersampling) the majority class**
  - Sample the data records from majority class
    - Randomly
    - Near miss examples
    - Examples far from minority class examples (far from decision boundaries)
  - Introduce sampled data records into the original data set instead of original data records from the majority class
  - Usually results in a general loss of information and potentially overly general rules

* Ling, C., Li, C. Data mining for direct marketing: Problems and solutions, KDD-98.
** Kubat M., Matwin, S., Addressing the Curse of Imbalanced Training Sets: One-Sided Selection, ICML 1997.

AHPCRC

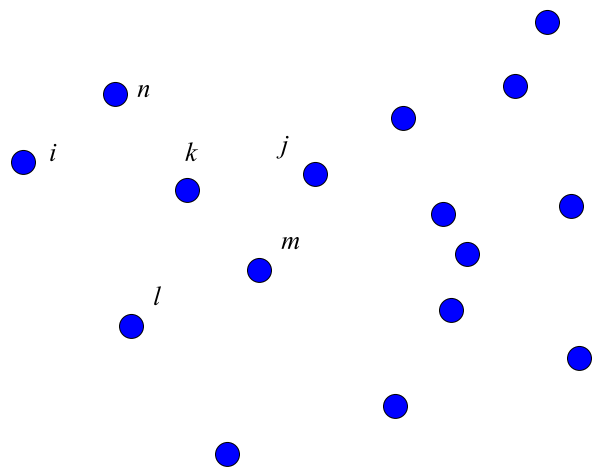
Manipulating Data Records – generating examples

- SMOTE (Synthetic Minority Over-sampling TEchnique)* - over-sampling the rare (minority) class by synthetically generating the minority class examples
- When generating artificial minority class example, distinguish two types of features
  - Continuous features
  - Nominal (Categorical) features
- Generating artificial anomalies** [Fan 2001]
  - Artificial anomalies are generated around the edges of the sparsely populated data regions

* N. Chawla, K. Bowyer, L. Hall, P. Kegelmeyer, SMOTE: Synthetic Minority Over-Sampling Technique, JAIR, vol. 16, 321-357, 2002.
** W. Fan et al, Using Artificial Anomalies to Detect Unknown and Known Network Intrusions, IEEE ICDM 2001

AHPCRC

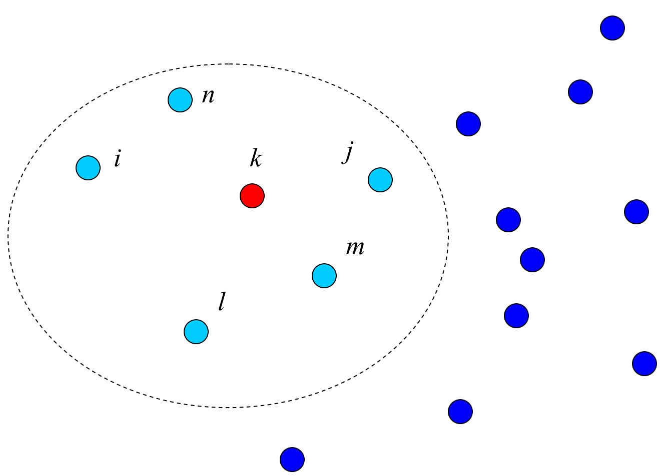
SMOTE Technique for Continuous Features



Blue points corresponds to minority class examples

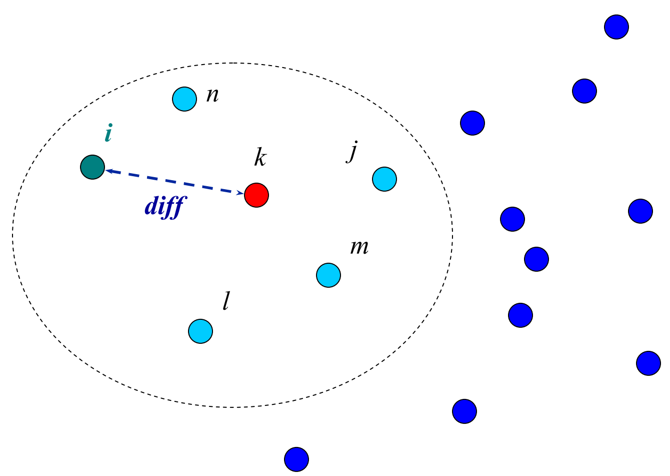
* N. Chawla, K. Bowyer, L. Hall, P. Kegelmeyer, SMOTE: Synthetic Minority Over-Sampling Technique, JAIR, vol. 16, 321-357, 2002.

SMOTE Technique for Continuous Features



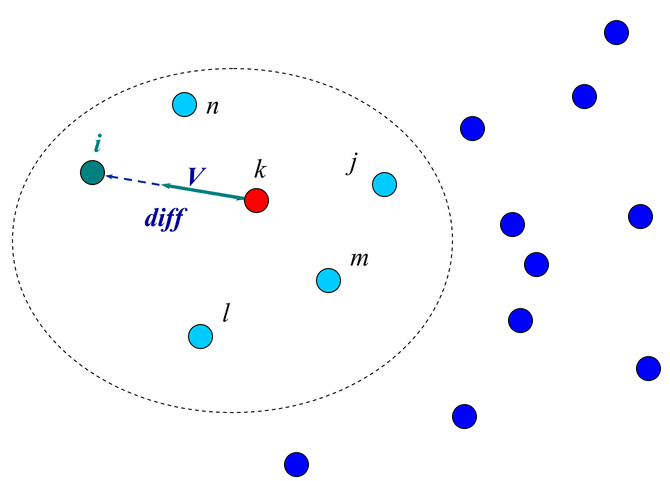
For each minority example *k* compute 5 nearest minority class examples {*i, j, l, n, m*}

SMOTE Technique for Continuous Features



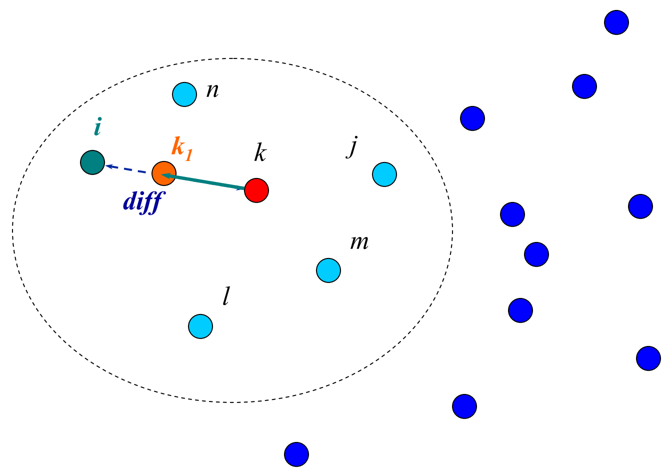
- Randomly choose an example out of 5 closest points
- Distance between the randomly chosen point ***i*** and the current point ***k*** is ***diff***

SMOTE Technique for Continuous Features



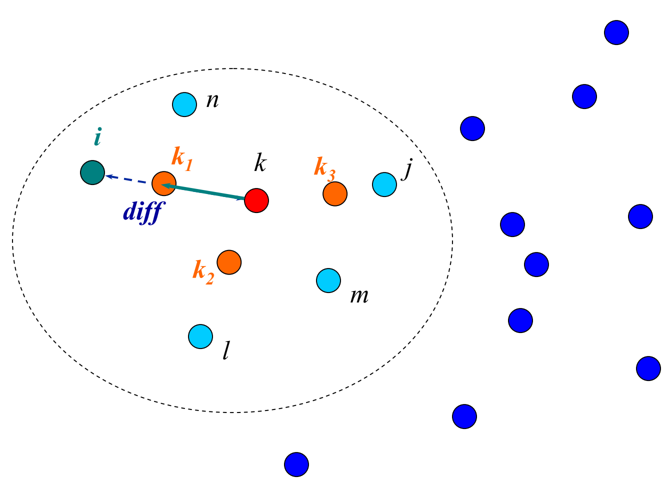
Randomly generate a number, such that it represents a vector ***V*** that has length that is less then ***diff***

SMOTE Technique for Continuous Features



Synthetically generate event k_1 based on the vector V, such that k_1 lies between point k and point i

SMOTE Technique for Continuous Features



After applying SMOTE 3 times (SMOTE parameter = 300%) data set may look like as the picture above

SMOTE Technique for Nominal Features

- For each minority example compute *k* nearest neighbors using Value Difference Metric (VDM)

$$\delta(V_1,V_2) = \sum_{i=1}^{n} \left| \frac{C_{1i}}{C_1} - \frac{C_{2i}}{C_2} \right|^p$$

V_1, V_2 **– corresponding feature values**

  - C_1 – total number of occurrences of V_1
  - C_{1i} – total number of occurrences of V_1 for class *i*
  - *n* – number of classes, *p* – constant (usually 1)
- Create a synthetic example by taking a majority vote among the *k* nearest neighbors

Example: E_1 = A B C D E

2 nearest neighbors are:

E_2 = A F C G N

E_3 = H B C D N

E_{smote} = A B C D N

Generating artificial anomalies*

- For sparse regions of data generate more artificial anomalies than for the dense data regions
  - For each attribute value *v* generate [(# of occurrence for most frequent value) – (# of occurrences for this value)] anomalies ($v_a \neq v$, other attributes random from data set)

Values of Attribute *i*	Number of occurrences	Number of generated examples
A	1000	-
B	100	900
C	10	990

- Filter artificial anomalies to avoid collision with known instance
- Use RIPPER to discover rule sets
- Pure anomaly detection vs. combined misuse and anomaly detection

* W. Fan et al, Using Artificial Anomalies to Detect Unknown and Known Network Intrusions, IEEE ICDM 2001.

*Design of New Algorithms: SHRINK**

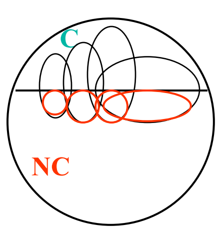
- **SHRINK insists that a mixed region is labeled as positive (rare) class, whether positives examples prevail in that region or not**
  - **Focus on searching best positive region (with maximum ratio positive examples to negative examples)**
    - **System is restricted to search for a single region to be labeled as positive**
  - **Induce classifier with low complexity**
    - **Classifier will be represented by the network of tests**
    - **Tests on numeric attributes have the form $x_i \in$ [min a_i, max a_i], Boolean x_i = 0 v 1**
    - **h_i - output of *i*-th test, $h_i = 1$ if the test suggests a positive label, $h_i = -1$ otherwise.**
    - **Example is classified positive if $\sum h_i \cdot w_i > \theta$, w_i - weight of *i*-th test** $\omega_i = \log\frac{e_i}{1-e_i}, e_i = l_i \cdot p^T$
    - **Remove min a_i or max a_i whichever reduces more radically # of negative examples and results in better g-mean score** $g-mean = \sqrt{acc^+ \cdot acc^-}$
    - **Find the interval with the maximum *g*-mean score and select it as the test**
    - **Tests with gi > 0.5 are discarded**

AHPCRC

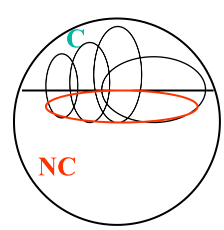
* M. Kubat,R. Holte, S. Matwin, Learning when Negative Examples Abound, ECML-97

*Design of New Algorithms: PN-rule Learning**

- *P-phase*:
  - **cover most of the positive examples with high support**
  - **seek good recall**
- *N-phase*:
  - **remove FP from examples covered in P-phase**
  - **N-rules give high accuracy and significant support**



Existing techniques can possibly learn erroneous small signatures for absence of C



PNrule can learn strong signatures for presence of NC in *N-phase*

AHPCRC

* M. Joshi, et al., PNrule, Mining Needles in a Haystack: Classifying Rare Classes via Two-Phase Rule Induction, ACM SIGMOD 2001

Design of New Algorithms: CREDOS*

- Ripple Down Rules (RDRs) offer a unique tree based representation that generalizes the decision tree and DNF rule list models and specializes a generic form of multi-phase PNrule model
- First use ripple down rules to overfit the training data
  - Generate a binary tree where each node is characterized by the rule R_h, a default class and links to two child subtrees
  - Grow the RDS structure in a recursive manner
  - Induces rules at a node
- Prune the structure in one pass to improve generalization using Minimum Description Length (MDL) principle
  - Different mechanism from decision trees

AHPCRC

* M. Joshi, et al., CREDOS: Classification Using Ripple Down Structure (A Case for Rare Classes), SIAM International Conference on Data Mining, (SDM'04), 2004.

Case specific feature/rule weighting

- Case specific feature weighting*
  - Information gain case-based learning (IG-CBL) algorithm
    - Create decision tree for the learning task
    - Compute the feature weights
      - $w_f = IG(f)$ if f is in the generated decision tree (IG – information gain), otherwise $w_f = 0$
    - Testing phase:
  - For each rare class test example replace global weight vector with dynamically generated weight vector that depends on the path taken by that test example
- Case specific rule weighting**
  - LERS (Learning from Examples based on Rough Sets) algorithm uses modified "bucket brigade" algorithm to create certain and possible rules
    - Strength – how well the rule performed during the training phase
    - Support – sum of scores of all the matching rules from the concept
  - Increase the rule strength for all rules describing the rare class

AHPCRC

* C. Cardie, N. Howe, Improving Minority Class Prediction Using Case specific feature weighting, ICML-1997.

** J. Grzymala et al, An Approach to Imbalanced Data Sets Based on Changing Rule Strength, AAAI Workshop on Learning from Imbalanced Data Sets, 2000.

Standard Boosting - Background

- Manipulates training examples to generate multiple classifiers
- Proceeds in a series of rounds
- Maintains a D_t - distribution of weights w_t over the training examples
- $D_{t+1}(i) = (D_t(i)/Z_t) \cdot e^{(-\alpha_t y_i h_t(x_i))}$ $\alpha_t = \frac{1}{2} \cdot \ln\left(\frac{1+r_t}{1-r_t}\right)$ - importance weight
  where Z_t is a normalization constant chosen such that D_{t+1} is a distribution
- In each round t, the learning algorithm is invoked to output a classifier C_t

AHPCRC

Boosting – Combining Classifiers

- The error of classifier C_t (h_t) is used to update the sampling weights of the training examples
  - Misclassified examples – higher weights
  - Correctly classified examples – smaller weights
- The final classifier is constructed as a weighted vote of individual classifiers
- Each classifier C_t (h_t) is weighted by α_t according to its accuracy on the entire training set

$$H(x) = sign\left(\sum_{t=1}^{M} \alpha_t h_t(x)\right)$$

AHPCRC

Boosting Based Algorithms - SMOTEBoost

- **SMOTEBoost embeds SMOTE technique in the boosting procedure**
  - **Utilize SMOTE for improving the prediction of the minority class**
  - **Utilize boosting for improving general performance**
- **After each boosting round, SMOTE is used to create new synthetic examples from the minority class**
- **SMOTEBoost indirectly gives higher weights to misclassified minority (rare) class examples**
- **Constructed classifiers are more diversed**

AHPCRC

* N. Chawla, A. Lazarevic, et al, SMOTEBoost: Improving the Prediction of Minority Class in Boosting, PKDD 2003.

Boosting Based Algorithms - SLIPPER

- **SLIPPER builds rules only for positive class**
  - **The only rule that predicts the negative class is a single default rule**
- **The weak hypotheses $h_t(x)$ used here are rules**
  - **Force the weak hypothesis based on a rule R to abstain (vote with confidence 0) on all instances not satisfied by R ($x \notin R$) by setting the prediction $h(x)$ for $x \notin R$ to 0**
  - **Force the rule to predict with the same confidence C_R on every $x \in R$, for t-th rule R_t, $\alpha_t h_t(x) = C_{Rt}$**

$$h_t(x) = \begin{cases} C_{R_t} & \text{if } x \in R_t \\ 0 & \text{otherwise} \end{cases} \qquad H(x) = sign\left(\sum_{R_t : x \in R_t} C_{R_t} \right)$$

AHPCRC

* W. Cohen, Y. Singer, A Simple, Fast and Effective Rule Learner, Annual Conference of American Association for Artificial Intelligence, 1999.

Boosting Based Algorithms - RareBoost

- **RareBoost-1**
  - Modify boosting by choosing different weight update factors for positives (TP, FP) and negatives (TN, FN) $\alpha_t^{\,p} = (1/2)\cdot\ln(TP_t / FP_t)$ $\alpha_t^{\,n} = (1/2)\cdot\ln(TN_t / FN_t)$
    - To achieve good recall for class C, distinguish between FN from TP
    - To achieve good precision for class C, distinguish between TP from FP

$$H(x) = sign\left(\sum_{h_t(x)\geq 0} \alpha_t^p h_t(x) + \sum_{h_t(x)<0} \alpha_t^n h_t(x) \right)$$

- **RareBoost-2**
  - Build rules for both the classes in every iteration
  - Modify SLIPPER by choosing between C and NC model based on whichever minimizes corresponding Z_t value
    - SLIPPER-C chooses between C and default
    - SLIPPER-NC chooses between NC and default

AHPCRC

* M. Joshi, et al., Predicting Rare Classes: Can Boosting Make Any Weak Learner Strong?, KDD 2002.

Cost Sensitive Classification

- Learning to minimize the expected cost of misclassifications
- Most classification learning algorithms attempt to minimize expected number of misclassification errors
- In many applications, different kinds of classification errors have different costs, so we need cost-sensitive methods
  - Intrusion Detection
    - False positive: security analysts waste time to analyze normal behavior
    - False negative: failure to detect intrusion could cause big damages
  - Fraud Detection
    - False positive: resources wasted investigating non-fraud
    - False negative: failure to detect fraud could be very expensive
  - Medical Diagnosis: Cost of false negative error - Postponed treatment or failure to treat; death or injury

AHPCRC

Cost Matrix

- *C(i,j)* = cost of predicting class *i* when the true class is *j*
- Example: Misclassification Costs Diagnosis of Cancer

Predicted State of Patient	True State of Patient	
	Positive - 0	Negative - 1
Positive – 0	C(0,0) = 1	C(0,1) = 1
Negative - 1	C(1,0) = 100	C(1,1) = 0

- If M is the confusion matrix for a classifier: *M(i,j)* is the number of test examples that are predicted to be in class *i* when their true class is *j*
- Expected misclassification cost is Hadamard product of *M* and *C* divided by the number of test examples *N*:

$$\frac{1}{N}\sum_{i,j=1}^{N} M(i,j)\cdot C(i,j)$$

AHPCRC

Cost Sensitive Classification Techniques

- Two Learning Problems
  - Cost *C* known at learning time
  - Cost *C* not known at learning time (only becomes available at classification time)
    - Learned classifier should work well for a wide range of costs
- Learning with known cost *C*
  - Find a classifier *h* that minimizes the expected misclassification cost on new data set points
  - Two basic strategies
    - Modify the inputs to the learning algorithm to reflect cost *C*
    - Incorporate cost *C* into the learning algorithm

AHPCRC

Cost Sensitive Classification Techniques

- Modifying inputs of learning algorithm
  - If there are 2 classes and the cost of a false positive is λ times larger than the cost of a false negative, put a weight of λ on each negative training example
  - $\lambda = C(1,0) / C(0,1)$
  - Then apply the learning algorithm as before
- Setting λ by class frequency (less frequent class has higher cost)
  - $\lambda \sim 1/n_k$, n_k - number of training examples from class k
- Setting λ by cross-validation
- Setting λ according to class frequency is cheaper and gives the same results as setting λ by cross validation

AHPCRC

Cost Sensitive Learning: MetaCost

- MetaCost wraps a "cost-minimizing" procedure around an arbitrary classifier, so that the classifier effectively minimizes cost, while seeking to minimize zero-one loss
- Bayes optimal prediction for x is class i that minimizes
  - $R(i|x) = \Sigma\, P(j|x)\, C(i,j)$
  - Conditional risk $R(i|x)$ is expected cost of predicting that x belongs to class i
  - Both $C(i,j)$ and $P(j|x)$ together with the above rule mean that the example space X can be partitioned into j regions, such that class j is the best (optimal) prediction in region j
- MetaCost is based on estimating class probabilities $P(j|x)$
- Standard probability estimation methods can determine these probabilities, but require machine learning bias of both the classifier and the density

AHPCRC

* P. Domingos, MetaCost: A general Method for making Classifiers Cost-Sensitive, KDD 1999.

Cost Sensitive Learning: MetaCost

- MetaCost estimates the class probabilities by learning multiple classifiers and for each example it uses the class's fraction of the total vote as an estimate
  - Method for estimating probabilities for modern unstable learners
- *Bagging* – given a training set *S*, a "bootstrap" sample of it is constructed by taking |*S*| samples with replacement
- MetaCost is different from bagging, where it uses smaller number of examples in each sample, which allows it to be more efficient
- While estimating class probabilities, MetaCost allows taking all the models generated (samples) into consideration, or only those samples
- The first type has a lower variance, because of its larger samples, and the second has lower statistical bias

AHPCRC

* P. Domingos, MetaCost: A general Method for making Classifiers Cost-Sensitive, KDD 1999.

Cost Sensitive Classification Techniques

- Modifying learning algorithm - Incorporating cost *C* into the learning algorithm
  - Cost-Sensitive Boosting
    - AdaCost*
    - CSB**
    - SSTBoost***
  - Cost *C* can be incorporated directly into the error criterion when training neural networks (Kukar & Kononenko, 1998)

AHPCRC

* W. Fan, S. Stolfo, J. Zhang, and P. Chan, Adacost: Misclassification cost-sensitive boosting, ICML 1999.

** K. M. Ting, A Comparative Study of Cost-Sensitive Boosting Algorithms, ICML 2000.

*** S. Merler, C. Furlanello, B. Larcher, and A. Sboner. Automatic model selection in cost-sensitive boosting, *Information Fusion*, 2003

Cost Sensitive Boosting

- Training examples of the form (x_i, y_i, c_i), where c_i is the cost of misclassifying x_i
- Boosting: $w_i := w_i * \exp(-\alpha_t\, y_i\, h_t(x_i))/Z_t$
- AdaCost [Fan et al., 1999]
  - $w_i := w_i * \exp(-\alpha_t\, y_i\, h_t(x_i)\, \beta_i)/Z_t$
  - $\beta_i = ½ * (1 + c_i)$ if error
    $= ½ * (1 - c_i)$ otherwise
- Cost-Sensitive Boosting (CSB) [Ting, 2000]
  - $w_i := \beta_i\, w_i * \exp(-\alpha_t\, y_i\, h_t(x_i))/Z_t$
  - $\beta_i = c_i$ if error
    $= 1$ otherwise
- SSTBoost [Merler et al., 2003]
  - $w_i := w_i * \exp(-\alpha_t\, y_i\, h_t(x_i)\, \beta_i)/Z_t$
  - $\beta_i = c_i$ if error
  - $\beta_i = 2 - c_i$ otherwise
  - $c_i = w$ for positive examples; $1 - w$ for negative examples

AHPCRC

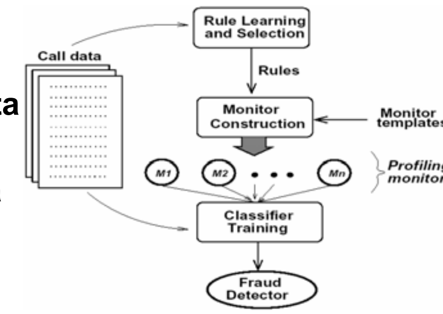
Learning with Unknown C

- Construct a classifier *h(x,C)* that can accept the cost function at run time and minimize the expected cost of misclassification errors with respect to cost *C*
- Approach: Learning to estimate *P(y|x)*
  - Probability Estimation Trees [Provost, Domingos 2000]
  - Bagged Probability Estimation Trees [Provost, Domingos 2000]
  - Lazy Option Trees [Friedman96, Kohavi97, Margineantu & Dietterich, 2001]
  - Bagged Lazy Option Trees [Margineantu, Dietterich 2001]

AHPCRC

Internally bias discrimination*

- A rule learning algorithm is used to uncover indicators of fraudulent behavior from a data set of customer transactions
- Indicators are used to create a set of monitors, which profile legitimate customer behavior and indicate anomalies



- Outputs of the monitors are used as features in a system that learns to combine evidence to generate high confidence alarms
- The system has been applied to the problem of detecting cellular cloning fraud

AHPCRC * T. Fawcett, F. Provost, Adaptive Fraud Detection, DMKD 1997.

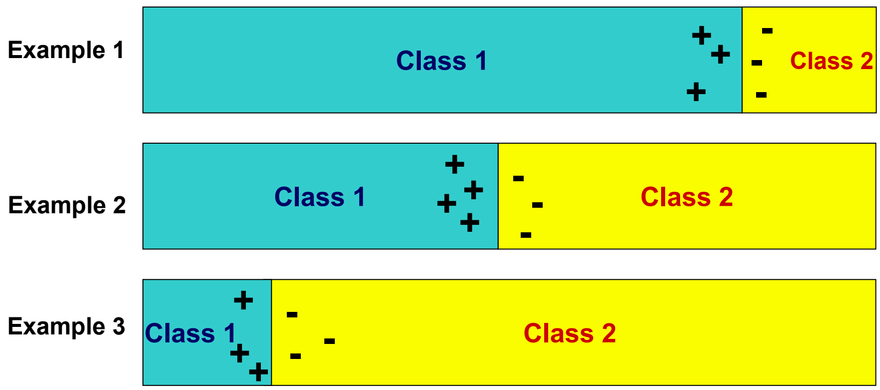
Selective Sampling based on Query Learning

- Active/Query Learning [Angluin 88]
  - Learner chooses examples on which the labels are requested
- Uncertainty Sampling [Lewis and Gale 94]
  - Learner queries examples for which its prediction so far is uncertain in order to maximize information gain
  - Example: Query by committee [Seung et al 92], which queries examples on which models obtained so far disagree on
  - Successfully applied to getting labeled data for text classification
- Selective sampling by query learning [Freund et al 93]
  - Given large number of (possibly unlabeled) data, uses only small subset of labeled data for learning
  - Successfully applied to mining very large data set [Mamitsuka and Abe 2000], even when labeled data are abound

AHPCRC

Query learning for imbalanced data set

- Use query learning to get more data for rare classes
- *Use selective sampling by query learning to get data near the decision boundary*



AHPCRC

* N. Abe, Sampling Approaches to Learning from Imbalanced Datasets, ICML Workshop on Imbalanced Data Sets II, 2003.

Sampling for Cost-sensitive Learning*

- Cost *C* depends on particular example x

	True = 0	True = 1
Predict = 0	$C(0,0,x)$	$C(0,1,x)$
Predict = 1	$C(1,0,x)$	$C(1,1,x)$

- Presents reduction of cost-sensitive learning to classification
  - Altering the original example distribution by multiplying it by a factor proportional to the relative cost of each example, makes any error minimizing classifier accomplish expected cost minimization on the original distribution
  - Proposes Costing (cost-sensitive ensemble learning)
- Empirical evaluation using benchmark data sets from targeted marketing domain
  - Costing has excellent predictive performance (w.r.t. cost minimization)
  - Costing is computationally efficient

AHPCRC

* B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting, IEEE ICDM 2003.

Association Rules for Rare Class

- **Emerging patterns***
  - For 2 classes C_1 and C_2 and support $s_i(X)$ of itemset X in class C_i growth rate is defined as $s_2(X)/s_1(X)$
  - Given a threshold $p > 1$, itemset X is p-emerging pattern (*EP*) if growth rate $\geq p$
  - Find *EPs* in rare class
  - Find values that have the highest growth rate from the major class to the rare class
  - Replace different attribute values in the original rare class *EPs* with the highest growth rate values
  - New generated *EPs* have a strong power to discriminate rare class from the major class

AHPCRC

* H. Alhammady, K. Rao, The Application of Emerging Patterns for Improving the Quality of Rare-class Classification, PAKDD 2004.

Temporal Analysis of Rare Events

- **Surprising patterns [Keogh et al, KDD02]**
  - A time series pattern *P*, extracted from database *X* is surprising relative to a database *R*, if the probability of its occurrence is greatly different to that expected by chance, assuming that *R* and *X* are created by the same underlying process.
- **Steps (TARZAN algorithm)**
  - Discretizing time-series into symbolic strings
    - Fixed sized sliding window
    - Slope of the best-fitting line
  - Calculate probability of any pattern, including ones we have never seen before using Markov models
  - For maintaining linear time and space property, they use suffix tree data structure
  - Computing scores by comparing trees between reference data and incoming information stream

AHPCRC

* E. Keogh, et al, Finding Surprising Patterns in a Time Series Database in Linear Time and Space, KDD 2002.

Temporal Analysis of Rare Events

- Learning to Predict Extremely Rare Events*
  - Genetic-based machine learning system that predicts events by identifying temporal and sequential patterns in data
- Temporal Sequence Associations for Rare Events**
- Predicting Rare Events In Temporal Domains***
  - Transform event prediction problem into a search for all patterns (event-sets) on the rare class exclusively
  - Discriminative power of patterns is validated against other classes
  - Patterns are then combined into a rule-based model for prediction

* G. Weiss, H. Hirsh, Learning to Predict Extremely Rare Events, AAAI Workshop on Learning from Imbalanced Data Sets, 2000.

** J. Chen, et al, Temporal Sequence Associations for Rare Events, PAKDD 2004

*** R. Vilalta, Predicting Rare Events In Temporal Domains, IEEE ICDM 2002.

Using Clustering for Rare Class Problems*

- Distinguish between regular "between class imbalances" and "within-class imbalance", where a single class is composed of various sub-classes of different size
- The procedure
  - Use clustering to identify subclasses (subclusters)
  - Each subcluster of the majority class is resampled until it reaches the size of the biggest subcluster in the majority class
    - Denote size of the new majority class as: Size_of_new_majority_class
  - Each subcluster of the rare class is resampled until it reaches the size
    - $\text{Size\_of\_majority\_class} / N_{\text{subclusters\_in\_rare\_class}}$

* N. Japkowicz, Class Imbalances: Are We Focusing on the Right Issues, ICML Workshop on Imbalanced Data Sets II, 2003.

Case Studies

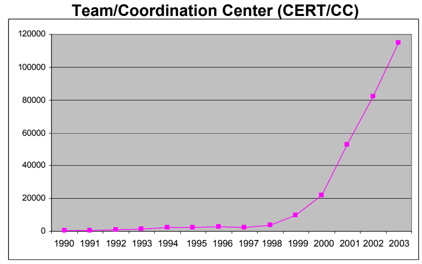
- Intrusion Detection
  - Network Intrusion Detection
  - Host based Intrusion Detection
- Fraud Detection
  - Credit Card Fraud
  - Insurance Fraud Detection
  - Cell Fraud Detection
- Medical Diagnostics
  - Mammogramy images
  - Health Care Fraud

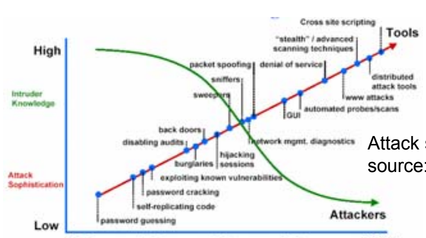
AHPCRC

Case Study: Data Mining in Intrusion Detection

- Due to the proliferation of Internet, more and more organizations are becoming vulnerable to cyber attacks
- Sophistication of cyber attacks as well as their severity is also increasing

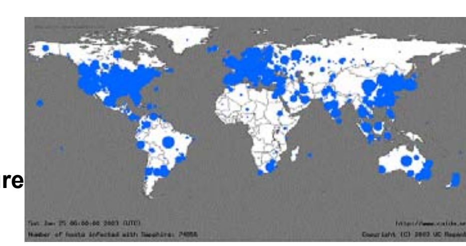
Incidents Reported to Computer Emergency Response Team/Coordination Center (CERT/CC)





Attack sophistication vs. Intruder technical knowledge, source: www.cert.org/archive/ppt/cyberterror.ppt

- Security mechanisms always have inevitable vulnerabilities
  - Firewalls are not sufficient to ensure security in computer networks
  - Insider attacks

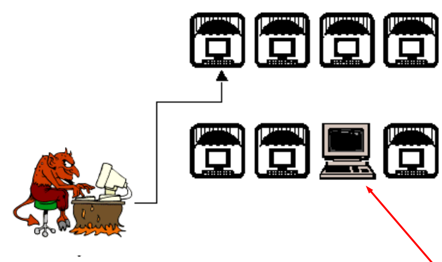


The geographic spread of Sapphire/Slammer Worm 30 minutes after release (**Source: www.caida.org**)

AHPCRC

What are Intrusions?

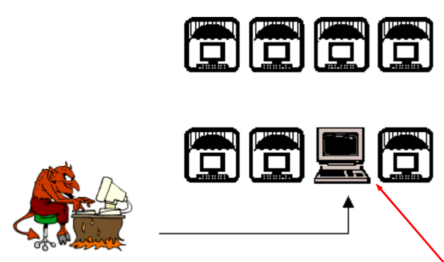
- Intrusions are actions that attempt to bypass security mechanisms of computer systems. They are usually caused by:
  - Attackers accessing the system from Internet
  - Insider attackers - authorized users attempting to gain and misuse non-authorized privileges
- Typical intrusion scenario



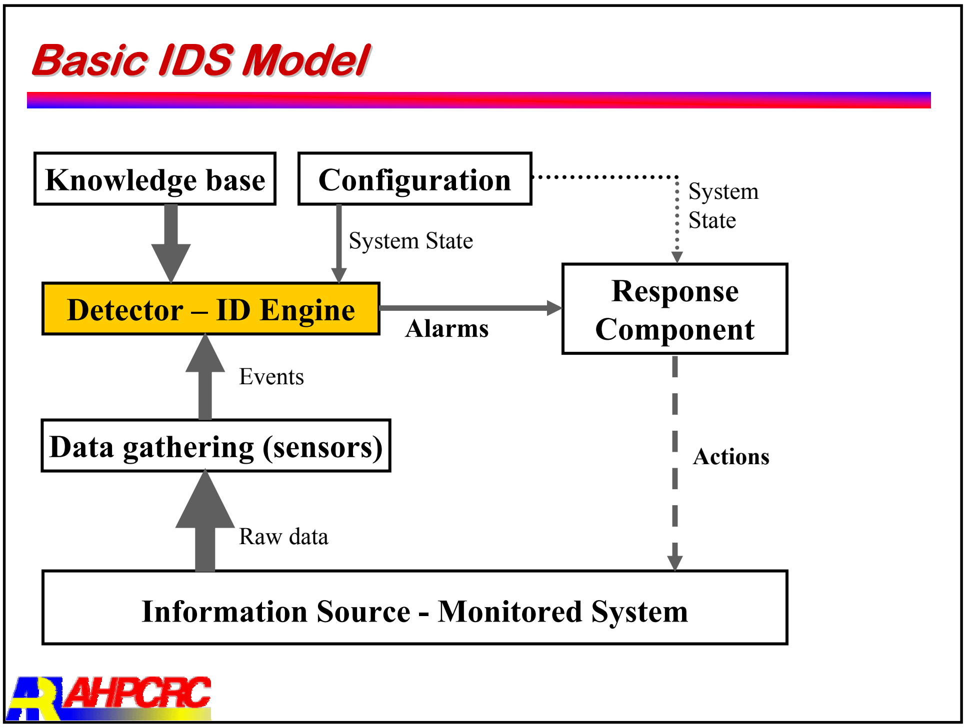
AHPCRC

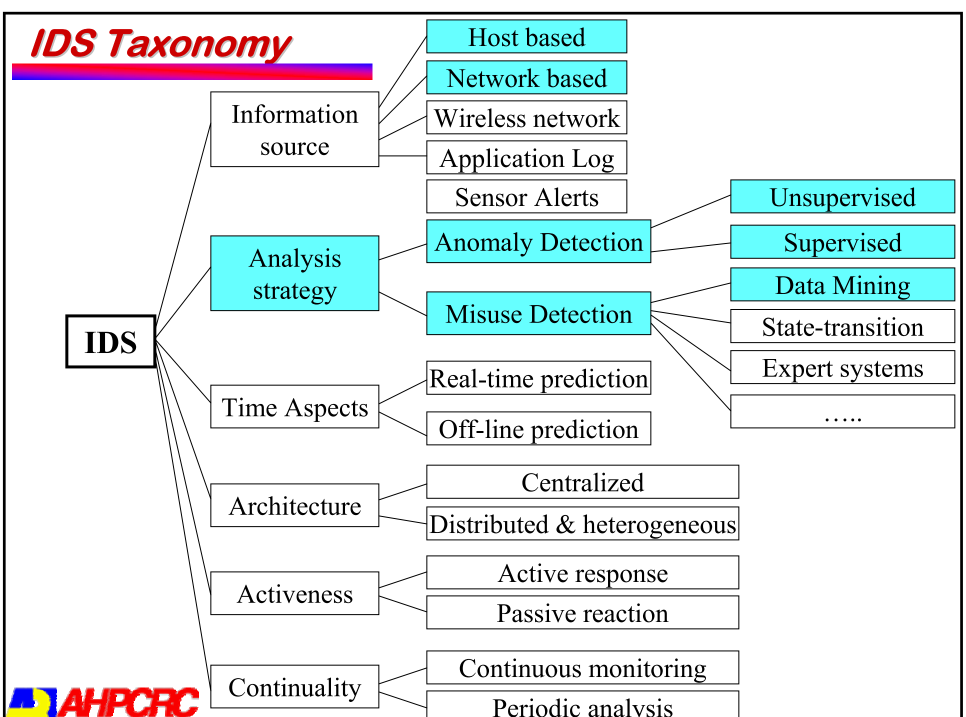
What are Intrusions?

- Intrusions are actions that attempt to bypass security mechanisms of computer systems. They are caused by:
  - Attackers accessing the system from Internet
  - Insider attackers - authorized users attempting to gain and misuse non-authorized privileges
- Typical intrusion scenario



AHPCRC


Basic IDS Model
Knowledge base
Configuration
System State
System State
Detector – ID Engine
Alarms
Response Component
Events
Data gathering (sensors)
Actions
Raw data
Information Source - Monitored System
AHPCRC


IDS Taxonomy
IDS
Information source
Host based
Network based
Wireless network
Application Log
Sensor Alerts
Analysis strategy
Anomaly Detection
Unsupervised
Supervised
Misuse Detection
Data Mining
State-transition
Expert systems
…..
Time Aspects
Real-time prediction
Off-line prediction
Architecture
Centralized
Distributed & heterogeneous
Activeness
Active response
Passive reaction
Continuality
Continuous monitoring
Periodic analysis
AHPCRC

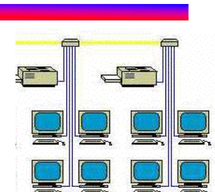
IDS - Analysis Strategy

- ***Misuse detection*** **is based on extensive knowledge of patterns associated with known attacks provided by human experts**
  - Existing approaches: pattern (signature) matching, expert systems, state transition analysis, data mining
  - Major limitations:
    - Unable to detect novel & unanticipated attacks
    - Signature database has to be revised for each new type of discovered attack
- ***Anomaly detection*** **is based on profiles that represent normal behavior of users, hosts, or networks, and detecting attacks as significant deviations from this profile**
  - Major benefit - potentially able to recognize unforeseen attacks.
  - Major limitation - possible high false alarm rate, since detected deviations do not necessarily represent actual attacks
  - Major approaches: statistical methods, expert systems, clustering, neural networks, support vector machines, outlier detection schemes

AHPCRC

Intrusion Detection

- **Intrusion Detection System**
  - combination of software and hardware that attempts to perform intrusion detection
  - raises the alarm when possible intrusion happens



- **Traditional intrusion detection system IDS tools (e.g. SNORT) are based on signatures of known attacks**
  - Example of SNORT rule (MS-SQL "Slammer" worm)

    any -> udp port 1434 (content:"|81 F1 03 01 04 9B 81 F1 01|"; content:"sock"; content:"send")

SNORT
www.snort.org

- **Limitations**
  - Signature database has to be manually revised for each new type of discovered intrusion
  - They cannot detect emerging cyber threats
  - Substantial latency in deployment of newly created signatures across the computer system
- **Data Mining can alleviate these limitations**

Data Mining for Intrusion Detection

- Increased interest in data mining based intrusion detection
  - Attacks for which it is difficult to build signatures
  - Attack stealthiness
  - Unforeseen/Unknown/Emerging attacks
  - Distributed/coordinated attacks
- Data mining approaches for intrusion detection
  - *Misuse detection*
    - Building predictive models from labeled labeled data sets (instances are labeled as "normal" or "intrusive") to identify known intrusions
    - High accuracy in detecting many kinds of known attacks
    - Cannot detect unknown and emerging attacks
  - *Anomaly detection*
    - Detect novel attacks as deviations from "normal" behavior
    - Potential high false alarm rate - previously unseen (yet legitimate) system behaviors may also be recognized as anomalies
  - *Summarization of network traffic*

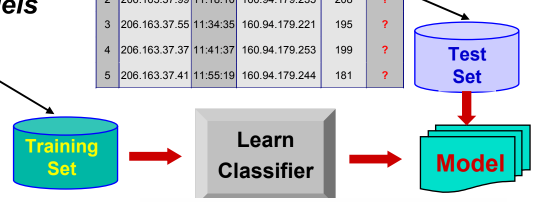
AHPCRC

Data Mining for Intrusion Detection

Tid	SrcIP	Start time	Dest IP	Dest Port	Number of bytes	Attack
1	206.135.38.95	11:07:20	160.94.179.223	139	192	No
2	206.163.37.95	11:13:56	160.94.179.219	139	195	No
3	206.163.37.95	11:14:29	160.94.179.217	139	180	No
4	206.163.37.95	11:14:30	160.94.179.255	139	199	No
5	206.163.37.95	11:14:32	160.94.179.254	139	19	Yes
6	206.163.37.95	11:14:35	160.94.179.253	139	177	No
7	206.163.37.95	11:14:36	160.94.179.252	139	172	No
8	206.163.37.95	11:14:38	160.94.179.251	139	285	Yes
9	206.163.37.95	11:14:41	160.94.179.250	139	195	No
10	206.163.37.95	11:14:44	160.94.179.249	139	163	Yes

Misuse Detection – Building Predictive Models

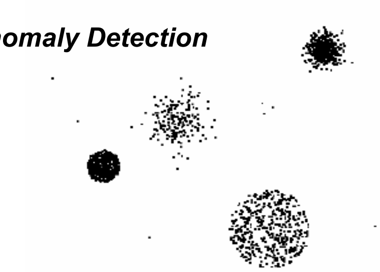
	categorical	temporal	categorical	continuous	class
Tid	SrcIP	Start time	Dest Port	Number of bytes	Attack
1	206.163.37.81	11:17:51	160.94.179.208	150	?
2	206.163.37.99	11:18:10	160.94.179.235	208	?
3	206.163.37.55	11:34:35	160.94.179.221	195	?
4	206.163.37.37	11:41:37	160.94.179.253	199	?
5	206.163.37.41	11:55:19	160.94.179.244	181	?



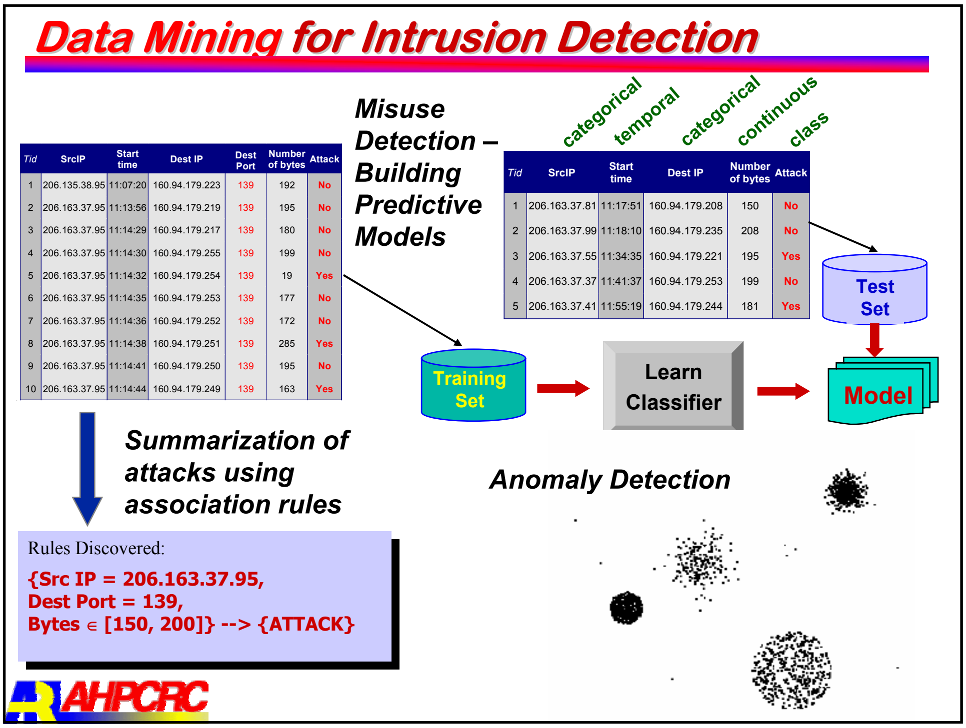
Summarization of attacks using association rules

Rules Discovered:
{Src IP = 206.163.37.95,
Dest Port = 139,
Bytes ∊ [150, 200]} --> {ATTACK}

Anomaly Detection



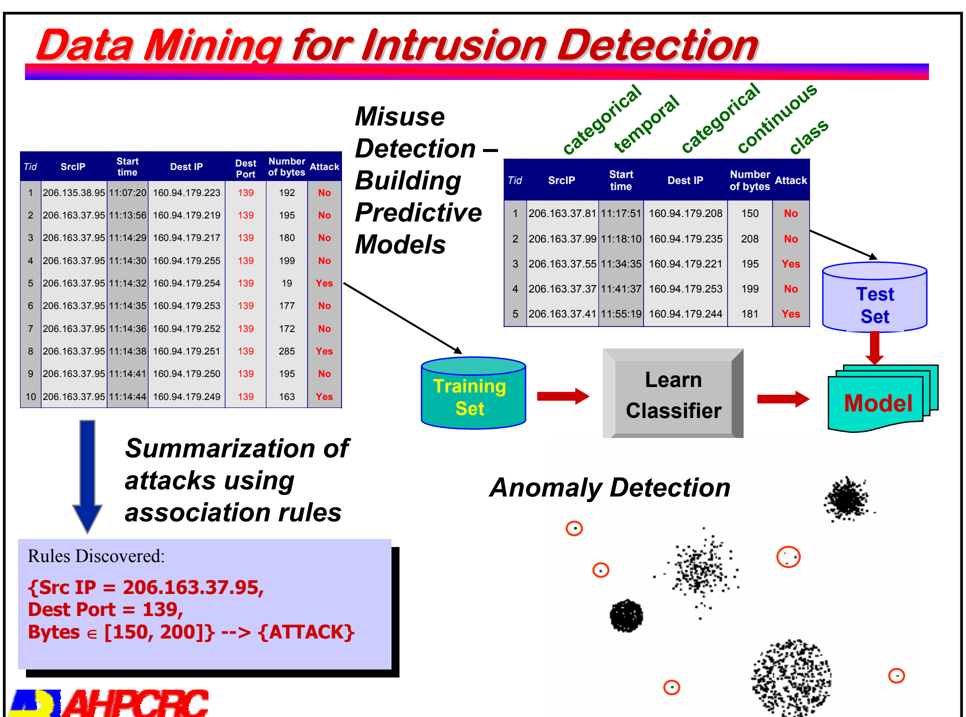
AHPCRC


Data Mining for Intrusion Detection
Misuse Detection – Building Predictive Models
categorical
temporal
categorical
continuous
class

Tid	SrcIP	Start time	Dest IP	Dest Port	Number of bytes	Attack
1	206.135.38.95	11:07:20	160.94.179.223	139	192	No
2	206.163.37.95	11:13:56	160.94.179.219	139	195	No
3	206.163.37.95	11:14:29	160.94.179.217	139	180	No
4	206.163.37.95	11:14:30	160.94.179.255	139	199	No
5	206.163.37.95	11:14:32	160.94.179.254	139	19	Yes
6	206.163.37.95	11:14:35	160.94.179.253	139	177	No
7	206.163.37.95	11:14:36	160.94.179.252	139	172	No
8	206.163.37.95	11:14:38	160.94.179.251	139	285	Yes
9	206.163.37.95	11:14:41	160.94.179.250	139	195	No
10	206.163.37.95	11:14:44	160.94.179.249	139	163	Yes

Tid	SrcIP	Start time	Dest IP	Number of bytes	Attack
1	206.163.37.81	11:17:51	160.94.179.208	150	No
2	206.163.37.99	11:18:10	160.94.179.235	208	No
3	206.163.37.55	11:34:35	160.94.179.221	195	Yes
4	206.163.37.37	11:41:37	160.94.179.253	199	No
5	206.163.37.41	11:55:19	160.94.179.244	181	Yes

Test Set
Training Set
Learn Classifier
Model
Summarization of attacks using association rules
Rules Discovered:
{Src IP = 206.163.37.95,
Dest Port = 139,
Bytes ∊ [150, 200]} --> {ATTACK}
Anomaly Detection
AHPCRC


Data Mining for Intrusion Detection
Misuse Detection – Building Predictive Models
categorical
temporal
categorical
continuous
class

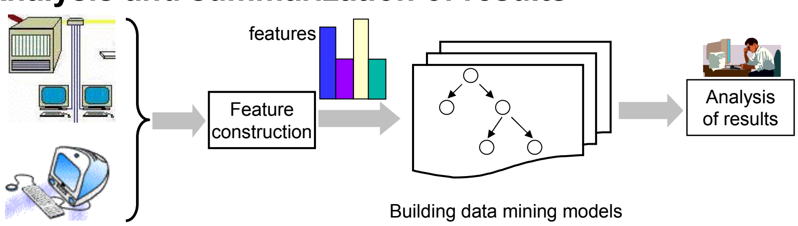
Tid	SrcIP	Start time	Dest IP	Dest Port	Number of bytes	Attack
1	206.135.38.95	11:07:20	160.94.179.223	139	192	No
2	206.163.37.95	11:13:56	160.94.179.219	139	195	No
3	206.163.37.95	11:14:29	160.94.179.217	139	180	No
4	206.163.37.95	11:14:30	160.94.179.255	139	199	No
5	206.163.37.95	11:14:32	160.94.179.254	139	19	Yes
6	206.163.37.95	11:14:35	160.94.179.253	139	177	No
7	206.163.37.95	11:14:36	160.94.179.252	139	172	No
8	206.163.37.95	11:14:38	160.94.179.251	139	285	Yes
9	206.163.37.95	11:14:41	160.94.179.250	139	195	No
10	206.163.37.95	11:14:44	160.94.179.249	139	163	Yes

Tid	SrcIP	Start time	Dest IP	Number of bytes	Attack
1	206.163.37.81	11:17:51	160.94.179.208	150	No
2	206.163.37.99	11:18:10	160.94.179.235	208	No
3	206.163.37.55	11:34:35	160.94.179.221	195	Yes
4	206.163.37.37	11:41:37	160.94.179.253	199	No
5	206.163.37.41	11:55:19	160.94.179.244	181	Yes

Test Set
Training Set
Learn Classifier
Model
Summarization of attacks using association rules
Rules Discovered:
{Src IP = 206.163.37.95,
Dest Port = 139,
Bytes ∊ [150, 200]} --> {ATTACK}
Anomaly Detection
AHPCRC

Data Preprocessing for Data Mining in ID

- **Converting the data from monitored system (computer network, host machine, …) into data (features) that will be used in data mining models**
  - **For misuse detection, labeling data examples into normal or intrusive may require enormous time for many human experts**
- **Building data mining models**
  - **Misuse detection models**
  - **Anomaly detection models**
- **Analysis and summarization of results**



AHPCRC

Data Sources in Network Intrusion Detection

- **Network traffic data is usually collected using "network sniffers"**
  - ***Tcpdump***

```
08:02:15.471817 0:10:7b:38:46:33 0:10:7b:38:46:33 loopback 60:
                0000 0100 0000 0000 0000 0000 0000 0000
                0000 0000 0000 0000 0000 0000 0000 0000
                0000 0000 0000 0000 0000 0000 0000
08:02:19.391039 172.16.112.100.3055 > 172.16.112.10.ntp:  v1 client strat 0 poll 0 prec 0
08:02:19.391456 172.16.112.10.ntp > 172.16.112.100.3055:  v1 server strat 5 poll 4 prec -16 (DF)
```

  - ***net-flow tools***
    - **Source and destination IP address, Source and destination ports, Type of service, Packet and byte counts, Start and end time, Input and output interface numbers, TCP flags, Routing information (next-hop address, source autonomous system (AS) number, destination AS number)**

```
0624.12:4:39.344  0624.12:4:48.292 211.59.18.101 4350  160.94.179.138 1433  6  2  3  144
0624.9:1:10.667   0624.9:1:19.635  24.201.13.122 3535  160.94.179.151 1433  6  2  3  132
0624.12:4:40.572  0624.12:4:49.496 211.59.18.101 4362  160.94.179.150 1433  6  2  3  152
```

- **Collected data are in the form of network connections or network packets (a network connection may contain several packets)**

AHPCRC

Feature Construction in Intrusion Detection

- Basic set of features (*start/end time, srcIP, dstIP, srcport, dstport, protocol, TCP flags, # of bytes, packets, …*)
- Three groups of features may be constructed (KDDCup 99):
  - "content-based" features within a connection
    - Intrinsic characteristics of data packets
    - number of failed logins, root logins, acknowledgments, directory creations, …)
  - time-based traffic features included number of connections or different services from the same source or to the same destination considering recent time interval (e.g.a few seconds)
    - Useful for detecting scanning activities
  - connection based features included number of connections from same source or to same destination or with the same service considering in last *N* connections
    - Useful for detecting SLOW scanning activities

MADAM ID - Feature Construction Example

- Example: "syn flood" patterns
  - (service=http, flag=SYN, victim), (service=http, flag=SYN, victim) → (service = http, SYN, victim) [0.9, 0.02, 2s]

dst … service … flag		
h1	http	S0
h1	http	S0
h1	http	S0
h2	http	S0
h4	http	S0
h2	ftp	S0

existing features useless

syn flood

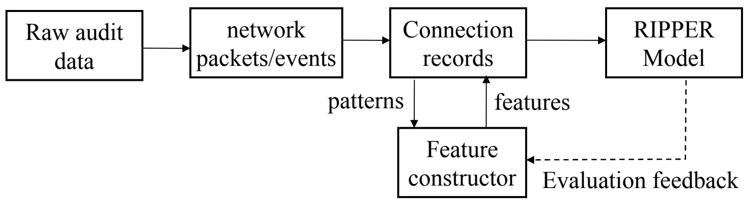
normal

dst … service … flag			%S0
h1	http	S0	70
h1	http	S0	72
h1	http	S0	75
h2	http	S0	0
h4	http	S0	0
h2	ftp	S0	0

construct features with high information gain

  - After 2 *http* packets with set SYN flag are sent to victim, in 90% of cases third connection is made within 2s. This happens in 2% of all connections (support = 2%)
  - Add features based on frequent episodes learned separately for normal data and attack data
    - count the connections in the past 2 seconds
    - count the connections with the same characteristics (http, SYN, victim) in the past 2 seconds
- Different features are found for different classes of attacks => different models for the classes

MADAM ID Workflow*



- **Association rules and frequent episodes are applied to network connection records to obtain additional features for data mining algorithms**
- **Apply RIPPER to labeled data sets and learn the intrusions**

AHPCRC

* W. Lee,S. Stolfo, Adaptive Intrusion Detection: a Data Mining Approach, *Artificial Intelligence Review,* 2000

MADAM ID - Cost-sensitive Modeling

- **A multiple-model approach:**
  - **Certain features are more costly to compute than others**
  - **Build multiple rule-sets, each with features of different cost levels**
  - **Cost factors: damage cost, response cost, operational cost (level 1-4 features)**
    - **Level 1: beginning of an event**
    - **Level 2: middle to end of an event**
    - **Level 3: at the end of an event**
    - **Level 4: at the end of an event, multiple events in a time window**
  - **Use cheaper rule-sets first, costlier ones later only for required accuracy**

AHPCRC

* W. Lee,et al., Toward Cost-Sensitive Modeling for Intrusion Detection and Response, *Journal of Computer Security,* 2002

ADAM*

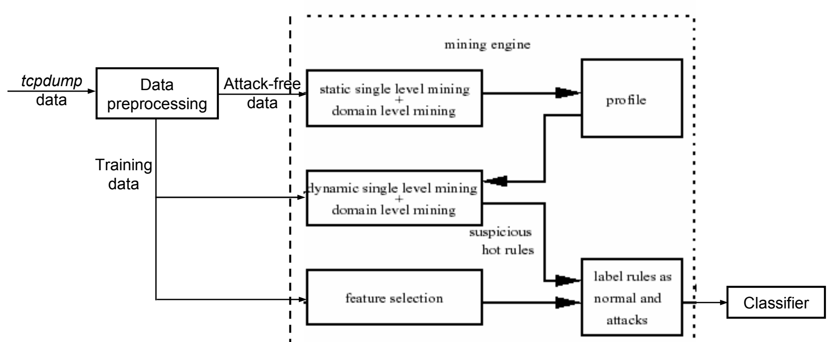
- Data mining testbed that uses combination of association rules and classification to discover attacks
- I phase: ADAM builds a repository of profiles for "normal frequent itemsets" by mining "attack-free" data
- II phase: ADAM runs a sliding window, incremental algorithm that finds frequent itemsets in the last N connections and compare them to "normal" profile
- Tunable: different thresholds can be set for different types of rules.
- Anomaly detection: first characterize normal behavior (profile), then flag abnormal behavior
- Reduced false alarm rate: using a classifier.

AHPCRC

* D. Barbara, et al., ADAM: A Testbed for Exploring the Use of Data Mining in Intrusion Detection. SIGMOD Record 2001.

ADAM: Training phase

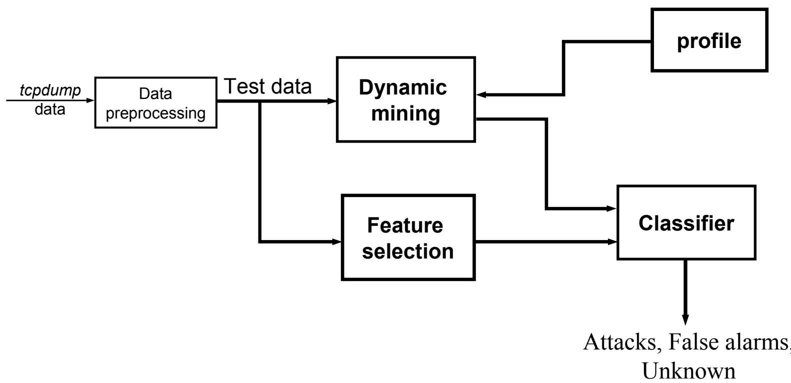
- Database of frequent itemsets for attack-free data is made
- For entire training data, find suspicious frequent itemsets that are not in the "attack-free" database
- Train a classifier to classify itemset as known attack, unknown attack or normal event



AHPCRC

ADAM: Phase of Discovering Intrusions

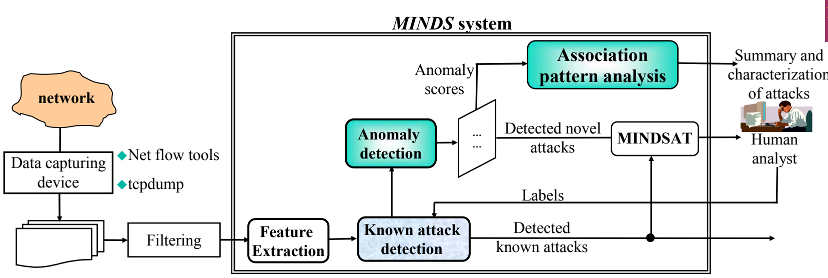
- Dynamic mining module produces suspicious itemsets from test data
- Along with features from feature selection module, itemsets are fed to classifier



AHPCRC

The MINDS Project*

- MINDS – **M**innesota **In**trusion **D**etection **S**ystem



- MINDS has been incorporated into the Interrogator architecture at the Army Research Labs (ARL) Center for Intrusion Monitoring and Protection (CIMP), where network data from multiple sensors is collected and analyzed.
- The MINDS is being used at University of Minnesota and at the ARL-CIMP to help analysts to detect attacks and intrusive behavior that cannot be detected using widely used intrusion detection systems, such as SNORT.

AHPCRC

* - Ertoz, L., Eilertson, E., Lazarevic, et al, The MINDS - Minnesota Intrusion Detection System, review for the book "Data Mining: Next Generation Challenges and Future Directions", AAAI/MIT Press.

Alternative Classification Approaches

- Fuzzy Data Mining in network intrusion detection (MSU)*
- Create fuzzy association rules only from normal data to learn "normal behavior"
  - For new audit data, create the set of fuzzy association rules and compute its similarity to the "normal" one
  - If similarity low ⇒ for new data generate alarm
- Genetic algorithms (GA) used to tune membership function of the fuzzy sets
  - Fitness – rewards for high similarity between normal and reference data, penalizing – high similarity between intrusion and reference data
- Use GA to select most relevant features

AHPCRC

* S. Bridges, R. Vaughn, Intrusion Detection Via Fuzzy Data Mining, 2000

Alternative Classification Approaches*

- Scalable Clustering Technique*
- Apply supervised clustering
  - For each point find nearest point and if belong to the same class, append to the same cluster, else create a new
- Classification
  - Class dominated in k nearest clusters
  - Weighted sum of distances to k nearest clusters
- Incremental clustering

- Distances: weighted Euclidean, Chi-square, Canbera

$$(d(x, L)) = \sum_{i=1}^{d} \frac{|x_i - L_i|}{x_i + L_i} C_i^2$$

L_i – i-th attribute of centroid of the cluster L

C_i – correlation between i-th attribute and the class

AHPCRC

* N. Ye, X. Li, A Scalable Clustering for Intrusion Signature Recognition, 2001.

Neural Networks Classification Approaches

- Neural networks (NNs) applied to host-based intrusion detection
  - Building profiles of users according to used commands
  - Building profiles of software behavior
- Neural networks for network-based intrusion detection
  - Hierarchical network intrusion detection
  - Multi-layer perceptrons (MLP)*
  - Self organizing maps (SOMs)*

AHPCRC

* J. Canady, J. Mahaffey, The Application of Artificial Neural Networks to Misuse Detection:Initial Results, 1998.

Statistics Based Outlier Detection Schemes

- *Packet level (PHAD) and Application level (ALAD) anomaly detection**
- PHAD (packet header anomaly detection) monitors Ethernet, IP and transport layer packet headers
  - It builds profiles for 33 different fields from these headers by looking attack free traffic and performs clustering (prespecifed # of clusters)
  - A new value that does not fit into any of the clusters (intervals), it is treated as a new cluster and closest two clusters are merged
  - The number of updates, *r*, is maintained for each field as well as the number of observations, *n*
  - Testing: For each new observed packet, if the value for some attribute does not fit into the clusters, anomaly score for that attribute is proportional to *n/r*
- ALAD uses the same method for anomaly scores, but it works only on TCP data and build TCP streams
  - It build profiles for 5 different features

AHPCRC

* M. Mahoney, P. Chan: Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks, 8th ACM KDD, 2002

NATE*

- **Low-cost anomaly detection algorithm**
  - It uses only header information
  - 5 features are used:
    - Counts of Push and Ack TCP flags (Syn, Fin, Reset and Urgent TCP flags are dropped due to their small variability)
    - Total number of packets
    - Number of bytes transferred for each packet (bytes / packet)
    - Percent of control packets per session
- **Builds clusters for normal behavior**
- **Multivariate normal distribution to model clusters**
- **For each test point compute Mahalanobis distance to every cluster**

AHPCRC

* C. Taylor and J. Alves-Foss. NATE - Network Analysis of Anomalous Traffic Events, A Low-Cost Approach, Proc. New Security Paradigms Workshop. 2001.

Summarization of Anomalous Connections*

- **MINDS example: January 26, 2003 (48 hours after the Slammer worm)**

score	srcIP	sPort	dstIP	dPort	protocol	flags	packets	bytes
37674.69	63.150.X.253	1161	128.101.X.29	1434	17	16	[0,2)	[0,1829)
26676.62	63.150.X.253	1161	160.94.X.134	1434	17	16	[0,2)	[0,1829)
24323.55	63.150.X.253	1161	128.101.X.185	1434	17	16	[0,2)	[0,1829)
21169.49	63.150.X.253	1161	160.94.X.71	1434	17	16	[0,2)	[0,1829)
19525.31	63.150.X.253	1161	160.94.X.19	1434	17	16	[0,2)	[0,1829)
19235.39	63.150.X.253	1161	160.94.X.80	1434	17	16	[0,2)	[0,1829)
17679.1	63.150.X.253	1161	160.94.X.220	1434	17	16	[0,2)	[0,1829)
8183.58	63.150.X.253	1161	128.101.X.108	1434	17	16	[0,2)	[0,1829)
7142.98	63.150.X.253	1161	128.101.X.223	1434	17	16	[0,2)	[0,1829)
5139.01	63.150.X.253	1161	128.101.X.142	1434	17	16	[0,2)	[0,1829)
4048.49	142.150.Y.101	0	128.101.X.127	2048	1	16	[2,4)	[0,1829)
4008.35	200.250.Z.20	27016	128.101.X.116	4629	17	16	[2,4)	[0,1829)
3657.23	202.175.Z.237	27016	128.101.X.116	4148	17	16	[2,4)	[0,1829)
3450.9	63.150.X.253	1161	128.101.X.62	1434	17	16	[0,2)	[0,1829)
3327.98	63.150.X.253	1161	160.94.X.223	1434	17	16	[0,2)	[0,1829)
2796.13	63.150.X.253	1161	128.101.X.241	1434	17	16	[0,2)	[0,1829)
2693.88	142.150.Y.101	0	128.101.X.168	2048	1	16	[2,4)	[0,1829)
2683.05	63.150.X.253	1161	160.94.X.43	1434	17	16	[0,2)	[0,1829)
2444.16	142.150.Y.236	0	128.101.X.240	2048	1	16	[2,4)	[0,1829)
2385.42	142.150.Y.101	0	128.101.X.45	2048	1	16	[0,2)	[0,1829)
2114.41	63.150.X.253	1161	160.94.X.183	1434	17	16	[0,2)	[0,1829)
2057.15	142.150.Y.101	0	128.101.X.161	2048	1	16	[0,2)	[0,1829)
1919.54	142.150.Y.101	0	128.101.X.99	2048	1	16	[2,4)	[0,1829)
1634.38	142.150.Y.101	0	128.101.X.219	2048	1	16	[2,4)	[0,1829)
1596.26	63.150.X.253	1161	128.101.X.160	1434	17	16	[0,2)	[0,1829)
1513.96	142.150.Y.107	0	128.101.X.2	2048	1	16	[0,2)	[0,1829)
1389.09	63.150.X.253	1161	128.101.X.30	1434	17	16	[0,2)	[0,1829)
1315.88	63.150.X.253	1161	128.101.X.40	1434	17	16	[0,2)	[0,1829)
1279.75	142.150.Y.103	0	128.101.X.202	2048	1	16	[0,2)	[0,1829)
1237.97	63.150.X.253	1161	160.94.X.32	1434	17	16	[0,2)	[0,1829)
1180.82	63.150.X.253	1161	128.101.X.61	1434	17	16	[0,2)	[0,1829)
1107.78	63.150.X.253	1161	160.94.X.154	1434	17	16	[0,2)	[0,1829)

Potential Rules:

1.

{Dest Port = 1434/UDP #packets ∈ [0, 2)} --> Highly anomalous behavior (Slammer Worm)

2.

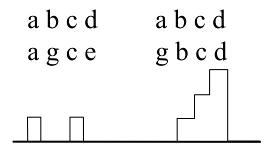
{Src IP = 142.150.Y.101, Dest Port = 2048/ICMP #bytes ∈ [0, 1829]} --> Highly anomalous behavior (ping – scan)

AHPCRC

* - Ertoz, L., Eilertson, E., Lazarevic, et al, The MINDS - Minnesota Intrusion Detection System, review for the book “Data Mining: Next Generation Challenges and Future Directions”, AAAI/MIT Press.

Profiling based anomaly detection

- Profiling methods are usually applied to host based intrusion detection where users, programs, etc. are profiled
  - Profiling sequences of Unix shell command lines*
    - Set of sequences (user profiles) reduced and filtered to reduce data set for analysis
    - Build Instance Based Learning (IBL) model that stores historic examples of "normal" data
      - Compares new data stream
      - Distance measure that favors long temporal similar sequences
      - Event sequences are segmented



  - Profiling users' behavior
    - Using neural networks

AHPCRC

* T. Lane, C. Brodley, Temporal Sequence Learning and Data Reduction for Anomaly detection, 1998.

Case Study: Data Mining in Fraud Detection

- Fraud detection in E-commerce/ business world
  - Credit Card Fraud
  - Internet Transaction Fraud / E-Cash fraud
  - Insurance Fraud and Health Care Fraud
  - Money Laundering
  - Telecommunications Fraud
  - Subscription Fraud / Identity Theft
- All major data miming techniques applicable to intrusion detection are also applicable to fraud detection domains mentioned above

AHPCRC

Data Mining in Credit Fraud Detection

The Washington Post

- Credit Card Companies Turn To Artificial Intelligence - "Credit card fraud costs the industry about a $billion a year, or 7 cents out of every $100 spent. But that is down significantly from its peak about a decade ago, Sorrentino says, in large part because of powerful technology that can recognize unusual spending patterns."

AHPCRC

Data Mining in Insurance Fraud Detection

BusinessWeek online

- Smart Tools
  - Banks, brokerages, and insurance companies have been relying on various AI tools for two decades. One variety, called a neural network, has become the standard for detecting credit-card fraud. Since 1992, neural nets have slashed such incidents by 70% or more for the likes of U.S. Bancorp and Wachovia Bank. Now, even small credit unions are required to use the software in order to qualify for debit-card insurance from Credit Union National Assn.

AAAI Press Room

- Innovative Use of Artificial Intelligence Monitoring NASDAQ for Potential Insider Trading and Fraud (September 17, 2003)

AHPCRC

Data Mining in Cell Fraud Detection

NewScientist.com

- Phone Friend - "More than 15,000 mobile phones are stolen each month in Britain alone. According to Swedish cellphone maker Ericsson, the fraudulent use of stolen mobiles means a loss of between two and five per cent of revenue for the network operators."

WIRELESSNEWSFACTOR.com

- Ericsson Enlists AI To Fight Wireless Fraud
  - Mobile network operators claim that 2 to 5 percent of total revenues are lost to fraud, and the problem is expected to worsen

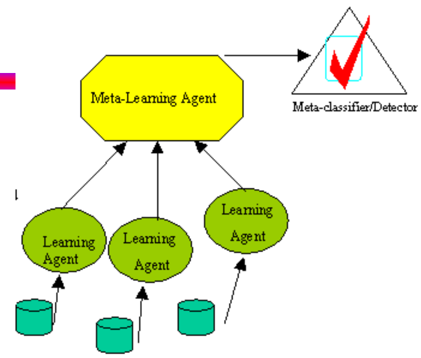
AHPCRC

Data Mining Techniques in Fraud Detection

- Basic approaches
  - JAM project [Chan99]
  - Neural networks [Brauser99, Maes00, Dorronsoro97]
  - Adaptive fraud detection [Fawcett97]
  - Account signatures [Cahill02]
  - Statistical Fraud Detection [Bolton 2002]
  - Fraud detection in mobile telecommunication networks [Burge96]
  - User profiling and classification for fraud detection in mobile telecommunication networks [Hollmen00]

AHPCRC

*JAM Project**

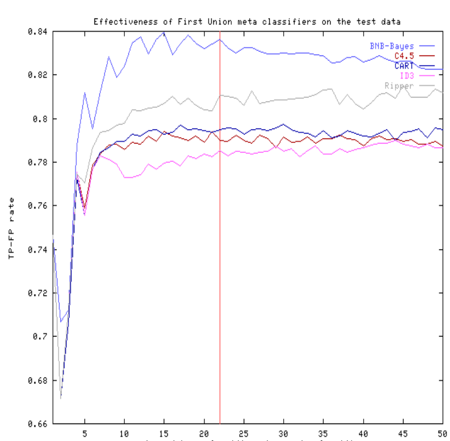


- **JAVA agents for meta-learning**
  - **Launch learning agents to remote data sites**
  - **Exchange remote classifiers**
  - **Local meta-learning agents produce meta-classifier**
  - **Launch meta-classifier to remote data sites**
- **Chase credit card (20% fraud), First Union credit card (15% fraud) – 500,000 data records**
- **5 base classifiers (Bayes, C4.5, CART, ID3, RIPPER)**
- **Select the classifier with the highest TP-FP rate on the validation set**

AHPCRC * JAM Project, http://www1.cs.columbia.edu/~sal/JAM/PROJECT/

JAM Project - Example

- **TP-FP rate vs. the number of classifiers**

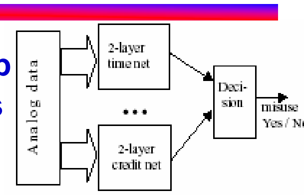


- **Input base classifiers:**
  First Union
- **Test data set:**
  First Union
- **Best meta classifier:**
  Naïve Bayes with 10-17 base classifiers.

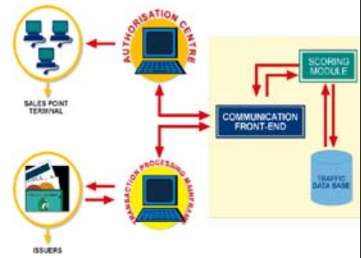
AHPCRC

Neural Networks in Fraud Detection*

- Use an expert net for each feature group (e.g. time,money) and group the experts together to form a common vote*



- Standard application of neural networks in credit card fraud detection **
- Use a non-linear version of Fisher's discriminant analysis, which separates a good proportion of fraudulent operations away from other closer to normal traffic



* R. Brause,T. Langsdorf, M. Hepp, Neural Data Mining for Credit Card Fraud Detection, 11th IEEE International Conference on Tools with Artificial Intelligence, 1999.

** S. Maes, et al, Credit Card Fraud Detection Using Bayesian and Neural Networks, NAISO Congress on NEURO FUZZY THECHNOLOGIES, 2002

*** J. Dorronsoro, F. Ginel, C. Sánchez, C. Santa Cruz, Neural Fraud Detection in Credit Card Operations, IEEE Trans. Neural Networks, 8 (1997), pp. 827-834

Fraud Detection in Telecommunication Networks

- Identify relevant user groups based on call data and then assign a user to a relevant group
  - call data is subsequently used in describing behavioral patterns of users
- Neural networks and probabilistic models are employed in learning these usage patterns from call data
- These models are used either to detect abrupt changes in established usage patterns or to recognize typical usage patterns of fraud

- J. Hollmen, User profiling and classification for fraud detection in mobile communications networks, PhD thesis, Helsinki University of Technology, 2000.
- P. Burge, J. Shawe-Taylor, Frameworks for Fraud Detection in mobile communications networks, 1996.

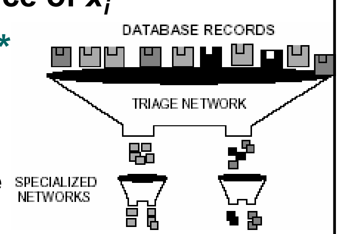
Case Study: Medical Diagnosis

- Cost of false positive error: Unnecessary treatment; unnecessary worry
- Cost of false negative error: Postponed treatment or failure to treat; death or injury

Event	Rate	Reference
Bleeding in outpatients on warfarin	6.7%/yr	Beyth 1998
Medication errors per order	5.3% (n=10070)	Bates 1995
Medication error per order	0.3% (n=289000)	Lesar 1997
Adverse drug events per admission	6.5%	Bates 1995
Adverse drug events per order	0.05% (n=10070)	Bates 1995
Adverse drug events per patient	6.7%	Lazarou 1998
Adverse drug events per patient	1.2%	Bains 1999
Adverse drug events in nursing home patients	1.89/100 pt-months	Gurwitz
Nosocomial infections in older hospitalized patients	5.9 to 16.9 per 1000 days	Rothchild 2000
Pressure ulcers	5%	Rothschild 2000

Data Mining in Medical Diagnosis

- Introducing noise to rare class examples*
  - For each attribute *i*, noisy example = $x_i + \varepsilon_i$, where $\varepsilon_i \approx N(0, \sigma^2_{noise}, \Sigma_q)$, Σ_q is *qxq* diagonal matrix *diag*$\{s_1^2, \ldots, s_q^2\}$, s_i^2 – sample variance of x_i
- Hierarchical neural nets (HNNs)**
  - HNNs are designed according to a divide-and-conquer approach
    - Triage networks are able to discriminate supersets that contain the infrequent pattern
    - These supersets are then used by specialized networks



* S. Lee, Noisy replication in skewed binary classification, Computational Statistics & Data Analysis August, 2000.

** L.Machado, Identification of Low Frequency Patterns in Backpropagation Neural Networks, Annual Symposium on Computer Applications in Medical Care, 1994.

Data Mining in Medical Diagnosis

- Spectrum and response analysis for neural nets*
- Data: polyproline type II (PPII) secondary structure
  - PPII structure is rare (1.26%)
  - Around an individual case (sequence) there is a hyper-sphere with a high probability area for the same secondary structure type
- Sample stratification schemes for neural networks (NNs)**
  1. stratifies a sample by adding up the weighted sum of the derivatives during the backward pass of training
  2. After training NNs with multiple sets of bootstrapped examples of the rare event classes and subsampled examples of common event classes, multiple voting for classification is performed

* M. Siermala, Local Prediction of Secondary Structures of Proteins from Viewpoints of Rare Structure, PhD Thesis, University of Tampere, May 2002.

** W. Choe et al, Neural network schemes for detecting rare events in human genomic DNA, Bioinformatics. 2000.

AHPCRC

Conclusions

- Data mining analysis of rare events requires special attention
- Many real world applications exhibit “needle-in-the-haystack” type of problem
- Current “state of the art” data mining techniques are still insufficient for efficient handling rare events
- Need for designing better and more accurate data mining models

AHPCRC

Links

- Intrusion Detection bibliography
  - www.cs.umn.edu/~aleks/intrusion_detection.html
  - www.cs.fit.edu/~pkc/id/related/index.html
  - www.cc.gatech.edu/~wenke/ids-readings.html
  - www.cerias.purdue.edu/coast/intrusion-detection/welcome.html
  - http://cnscenter.future.co.kr/security/ids.html
  - www.cs.purdue.edu/homes/clifton/cs590m/
  - www.cs.ucsb.edu/~rsg/STAT/links.html
- Fraud detection bibliography
  - www.hpl.hp.com/personal/Tom_Fawcett/fraud-public.bib.gz
  - http://dinkla.net !!!!!
  - http://www.aaai.org/AITopics/html/fraud.html
- Fraud detection solutions
  - www.kdnuggets.com/solutions/fraud-detection.html

AHPCRC

Questions?

Thank You!

Contact: aleks@cs.umn.edu

srivasta@cs.umn.edu

kumar@cs.umn.edu

AHPCRC