Systematic Construction of Anomaly Detection Benchmarks from Real Data

Andrew F. Emmott, Shubhomoy Das, Thomas Dietterich, Alan Fern, Weng-Keen Wong
Oregon State University
School of EECS
Corvallis, Oregon, USA
{emmott,dassh,tgd,afern,wong}@eecs.oregonstate.edu

ABSTRACT

Research in anomaly detection suffers from a lack of realistic and publicly-available problem sets. This paper discusses what properties such problem sets should possess. It then introduces a methodology for transforming existing classification data sets into ground-truthed benchmark data sets for anomaly detection. The methodology produces data sets that vary along three important dimensions: (a) point difficulty, (b) relative frequency of anomalies, and (c) clusteredness. We apply our generated datasets to benchmark several popular anomaly detection algorithms under a range of different conditions.

1. INTRODUCTION

Anomaly detection is an important task in many real-world applications, such as identifying novel threats in computer security [15, 23, 16, 21], finding interesting data points in scientific data [26], and detecting broken sensors (and other problems) in data sets [7]. Although a wide variety of anomaly detection algorithms have been developed and applied to these tasks [20, 11, 5, 31], a shortcoming of most published work is that there is no standard methodology for comparing anomaly detection methods. Instead, most published work either addresses data sets from specific applications or else employs synthetic data. This leads to three problems. First, with an application-specific data set, there is no independent way to assess the difficulty of the anomaly detection problem based on a standard set of properties of the data. Second, an application-specific data set limits us to the single data set from the application—there is no way to generate new data sets (aside from sub-sampling) that may differ in controlled ways. Third, with synthetic data sets, there is no real-world validity to the anomalies, so it is difficult to judge whether algorithms that work well on such simulated data will actually work well in a real-world setting.

In this paper, we attempt to address these shortcomings. In particular, our main contribution is to present a methodology for creating families of anomaly detection problems from real-world data sets. We begin in Section 2 by discussing the properties that benchmark data sets should possess in order to support rigorous evaluations of anomaly detection algorithms. Then in Section 3 we present the methodology that we have developed to create data sets with those properties. Section 4 describes an experiment in which we apply the methodology to benchmark several of the leading anomaly detection algorithms. Section 5 discusses the results of the experiment, and Section 6 presents our conclusions and suggestions for future work.

ODD'13, August 11th, 2013, Chicago, IL, USA.

2. REQUIREMENTS FOR ANOMALY DETECTION BENCHMARKS

The most common goal of anomaly detection is to raise an alarm when anomalous observations are encountered, such as insider threats [17], cyber attacks [15, 23, 16, 21], machine component failures [27, 28, 1], sensor failures [7], novel astronomical phenomena [26], or the emergence of cancer cells in normal tissue [22, 10]. In all of these cases, the underlying goal is to detect observations that are *semantically distinct* from normal observations. By this, we mean that the process that is generating the anomalies is different from the process that is generating the normal data points.

The importance of the underlying semantics suggests the first three requirements for benchmark datasets.

Requirement 1: Normal data points should be drawn from a real-world generating process. Generating data sets from some assumed probability distribution (e.g., a multivariate Gaussian) risks not capturing *any* real-world processes. Instead, as the field has learned from many years of experience with benchmark problems, it is important that the problems reflect the idiosyncrasies of real domains.

Requirement 2: The anomalous data points should also be from a real-world process that is semantically distinct from the process generating the normal points. The anomalous points should not just be points in the tails of the "normal" distribution. See, for example, Glasser and Lindauer's synthetic anomaly generator [8].

Requirement 3: Many benchmark datasets are needed. If we employ only a small number of data sets, we risk developing algorithms that only work on those problems. Hence,

we need a large (and continually expanding) set of benchmark data sets to ensure generality and prevent overfitting.

Requirement 4: Benchmark datasets should be characterized in terms of well defined and meaningful problem dimensions that can be systematically varied. An important goal for benchmarking is to gain insight into the strengths and weaknesses of the various algorithms. Ideally, we should identify those dimensions along which anomaly detection problems might vary and then generate benchmark data sets that vary these dimensions in a controlled fashion.

There is currently no established set of problem dimensions for anomaly detection, and we expect this set to evolve with experience. Here we propose four such dimensions: (a) point difficulty, (b) relative frequency, (c) semantic variation, and (d) feature relevance/irrelevance. The remainder of this section describes these in more detail.

Point difficulty measures the "distance" of an anomalous data point from the normal data points. We propose a point difficulty metric based on an oracle that knows the true generating processes underlying the "normal" and "anomalous" points. Using this knowledge, we suppose that the oracle can compute the probability $P(y = \text{normal}|x)$ that a data point x was generated by the "normal" distribution. The larger this value is for an anomalous point x, the more difficult it will be for an anomaly detection algorithm to discover that x is anomalous. One aspect of applying anomaly detection in adversarial settings (e.g., intrusion detection or insider threat detection) is that the adversaries try to blend in to the distribution of normal points.

Relative frequency is the fraction of the incoming data points that are (true) anomalies. The behavior of anomaly detection algorithms often changes with the relative frequency. If anomalies are rare, then methods that pretend that all training points are "normal" and fit a model to them may do well. If anomalies are common, then methods that attempt to fit a model of the anomalies may do well. In most experiments in the literature, the anomalies have a relative frequency between 0.01 and 0.1, but some go as high as 0.3 [14].

Semantic Variation is a measure of the degree to which the anomalies are generated by more than one underlying process. In this paper, we employ a measure of *clusteredness* as a proxy for this. If the anomalies are tightly clustered, then some anomaly detection algorithms will fail. For example, methods based on measures of local probability density will conclude that tightly clustered anomalies have high local density and hence are not anomalous.

Feature Relevance/Irrelevance. In applications, many candidate features are often available. However, many anomaly detection methods do not provide good feature selection mechanisms. Benchmark data sets should systematically vary the set of features to manipulate both the power of the relevant features and the number of irrelevant or "noise" features.

3. METHODOLOGY

We have developed a methodology that achieves most of the requirements listed above. To achieve the first three requirements, we develop 4,369 benchmark data sets by transforming 19 data sets chosen from the UC Irvine repository [2]. For each data set, we separate its data (e.g., the classes of a classification problem) into two sets: "normal" and "anomalous". This ensures that these data points are generated by distinct real-world processes rather than from synthesized distributions. To develop a measure of point difficulty, we fit a kernel logistic regression classifier to all of the available "normal" and "anomalous" data. This gives us an approximation to the oracle estimate of $P(y = \text{normal}|x)$. We can then manipulate the point difficulty of a benchmark data set by sampling the "anomalous" data points according to their point difficulty. It is easy to manipulate the relative frequency by varying the number of "anomalous" data points to include. We vary the degree of semantic variation by selecting data points that are either close together or far apart according to a simple distance metric. Our current methodology does not vary the feature relevance/irrelevance. This dimension is challenging to manipulate in a realistic manner, and we will investigate it further in future work.

3.1 Selecting Data Sets

To ensure reproducibility of our experiments, we only worked with data sets from the UCI data repository [2]. We selected all data sets that match the following criteria:

- *task*: Classification (binary or multi-class) or Regression. No Time-Series.
- *instances*: At least 1000. No upper limit.
- *features*: No more than 200. No lower limit.
- *values*: Numeric only. Categorical features are ignored if present. No missing values, except where easily ignored.

To ensure objectivity, we applied this fixed set of criteria rather than choosing data sets based on how well particular anomaly detection algorithms performed or based on our intuitions about which data sets might be better suited to creating anomaly detection problems.

If necessary, each data set was sub-sampled to 10,000 instances (while maintaining the class proportions for classification problems). Each feature was normalized to have zero mean and unit sample variance. We avoid time series because the majority of existing anomaly detection methods are based on models intended for independent and identically distributed data rather than for structured data such as time series data.

The 19 selected sets (grouped into natural categories) are the following:

- *binary classification*: MAGIC Gamma Telescope, MiniBooNE Particle Identification, Skin Segmentation, Spambase
- *multi-class classification*: Steel Plates Faults, Gas Sensor Array Drift, Image Segmentation, Landsat Satellite, Letter Recognition, Optical Recognition of Handwritten Digits, Page Blocks, Shuttle, Waveform, Yeast
- *regression*: Abalone, Communities and Crime, Concrete Compressive Strength, Wine, Year Prediction

3.2 Defining Normal versus Anomalous Data Points

A central goal of our methodology is that the "normal" and "anomalous" points should be produced by semantically distinct processes. To achieve this, we did the following.

For Irvine data sets that were already binary classification problems, we choose one class as "normal" and the other as "anomalous". Note that there is some risk that the "anomalous" points will have low semantic variation, since they all belong to a single class.

For multi-class data sets, we partition the available classes into two sets with the goal of maximizing the difficulty of telling them apart. Our heuristic procedure begins by training a Random Forest [3] to solve the multi-class classification problem. Then we calculate the amount of confusion between each class. For each data point x_i, the Random Forest computes an estimate of $P(\hat{y}_i|x_i)$, the predicted probability that x_i belongs to class $\hat{y}_i$. We construct a confusion matrix C in which cell $C[j,k]$ contains the sum of $P(\hat{y}_i = k|x_i)$ for all x_i whose true class $y_i = j$. We then define a graph in which each node is a class and each edge (between two classes j and k) has a weight equal to $C[j,k] + C[k,j]$. This is the (unnormalized) probability that a data point in class j will be confused with a data point in class k. We then compute the maximum weight spanning tree of this (complete) graph to identify a graph of "most-confusable" relationships between pairs of classes. We then two-color this tree so that no adjacent nodes have the same color. The two colors define the two sets of points. This approximately maximizes the confusions between "normal" and "anomalous" data points and also tends to make both the "normal" and "anomalous" sets diverse, which increases semantic variation in both sets.

For regression data sets, we compute the median of the regression response and partition the data into two classes by thresholding on this value. To the extent that low versus high values of the response correspond to different generative processes, this will create a semantic distinction between the "normal" and the "anomalous" data points. Points near the median will exhibit less semantic distinction, and they will also have high point difficulty.

3.3 Computing Point Difficulty

After reformulating all 19 Irvine tasks as binary classification problems, we simulate an omniscient oracle by applying Kernel Logistic Regression (KLR [12, 30, 13]) to fit a conditional probability model $P(y|x)$ to the data. Anomalies are labeled with $y = 0$ and normal points as $y = 1$. We then compute the logistic response for each candidate anomaly data point. Observe that points that are easy to discern from the "normal" class will have responses $P(y = 1|x)$ tending toward 0, while points that KLR confuses with the "normal" class will have responses above 0.5. Hence, for anomalous points, this response gives us a good measure of point difficulty.

For purposes of generating data sets, we assign each "anomalous" data point to one of four difficulty categories:

- *easy*: Difficulty score $\in (0, 0.1\overline{6})$
- *medium*: Difficulty score $\in [0.1\overline{6}, 0.\overline{3})$
- *hard*: Difficulty score $\in [0.\overline{3}, 0.5)$
- *very hard*: Difficulty score $\in [0.5, 1)$

Although we doubt that experiments derived from "very hard" candidate anomalies will resemble any real application domain, we decided to include them in our tests to see what impact they have on the results.

3.4 Semantic Variation and Clusteredness

Given a set of candidate "anomalous" data points, we applied the following algorithms to generate sets (of desired size) that are either widely dispersed or tightly clustered (as measured by Euclidean distance). To generate K dispersed points, we apply a facility location algorithm [9] to choose K points as the locations of the facilities. To generate K tightly clustered points, we choose a seed point at random and then compute the $K - 1$ data points that are closest to it in Euclidean distance. Note that when the point difficulty is constrained, then only candidate points of the specified difficulty are considered in this process. To quantify the clusteredness of the selected points, we measure the normalized *clusteredness*, which is defined as ratio of the sample variance of the "nominal" points to the sample variance of K selected "anomalous" points. When clusteredness is less than 1, the "anomalous" points exhibit greater semantic variance than the "normal" points. When clusteredness is greater than 1, the "anomalous" points are more tightly packed than the "normal" points (on average).

For purposes of analysis, we grouped the clusteredness scores into six qualitative levels: *high scatter* $(0, 0.25)$, *medium scatter* $[0.25, 0.5)$, *low scatter* $[0.5, 1)$, *low clusteredness* $[1, 2)$, *medium clusteredness* $[2, 4)$, and *high clusteredness* $[4, \infty)$.

3.5 Generating Benchmark Data Sets

To generate a specific data set, we choose a level of difficulty (easy, medium, hard, very hard), a relative frequency (0.001, 0.005, 0.01, 0.05, and 0.1), and a semantic variation setting (low or high). Then we apply the corresponding semantic variation procedure (with K set to achieve the desired relative frequency) to the set of available points of the desired difficulty level. For each combination of levels, we attempted to create 40 replicate data sets. However, when the number of candidate anomalous data points (at the desired difficulty level) is small, we limit the number of data sets to ensure that the replicates are sufficiently distinct. Specifically, let N be the number of available points. We create no more than $\lfloor N/K \rfloor$ replicates.

In total, from the 19 "mother" sets listed earlier, this methodology produced 4,369 problem set replicates, all of which we employed to test several statistical outlier detection algorithms.

4. ALGORITHMS

To simultaneously assess the effectiveness of our methodology and compare the performance of various statistical anomaly detection algorithms, we conducted an experimental study using several well-known anomaly detection algorithms. In this section, we describe each of those algorithms. For algorithms that required parameter tuning, we employed cross-validation (where possible) to find parameter values to maximize an appropriate figure of merit (as described below). In all cases, we made a good faith effort to maximize the performance of all of the methods. Some parameterization choices had to be made to ensure that the given algorithm implementation would return real-valued results.

4.1 One-Class SVM (ocsvm)

The One-Class SVM algorithm (Scholkopf et al. [24]) shifts the data away from the origin and then searches for a kernel-space decision boundary that separates fraction $1 - \delta$

of the data from the origin. We employ the implementation of Chang and Lin [6] available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`. For each benchmark, we employ a radial basis kernel and search parameter space until approximately 5% ($\delta = 0.05$) of the data lies outside the decision boundary in cross-validation. We would have preferred to use smaller values for δ, but OCSVM would not execute reliably for smaller values. The distance of a point from the decision boundary determines the anomaly score of that point.

4.2 Support Vector Data Description (svdd)

As proposed by Tax and Duin [25], Support Vector Data Description finds the smallest hypersphere (in kernel space) that encloses $1 - \delta$ of the data. We employed the `libsvm` implementation available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/` with a Gaussian radial basis function kernel. We search for parameters such that approximately 1% ($\delta = 0.01$) of the data lie outside the decision surface in cross validation. The distance of a point from the decision surface determines the anomaly score of that point.

4.3 Local Outlier Factor (lof)

The well-known Local Outlier Factor algorithm (Breunig, et al. [4]) computes the outlier score of a point x by computing its average distance to its k nearest neighbors. It normalizes this distance by computing the average distance of each of those neighbors to *their* k nearest neighbors. So, roughly speaking, a point is declared to be anomalous if it is significantly farther from its neighbors than they are from each other. We employed the `R` package `rlof` available at `http://cran.open-source-solution.org/web/packages/Rlof/`.

We chose k to be 3% of the data set. This was the smallest value for which LOF would reliably run on all data sets.

4.4 Isolation Forest (if) and Split-selection Criterion Isolation Forest (scif)

The Isolation Forest algorithm (Liu, et al. [18]) creates a forest of random axis-parallel projection trees. It derives a score based on the observation that points that become isolated closer to the root of a tree are easier to separate from the rest of the data and therefore are more likely to be anomalous. This method has a known weakness when the anomalous points are tightly clustered. To address this weakness, Liu, et al. [19] developed the Sparse-selection Criterion Isolation Forest. SCiForest subsamples the data points and features when growing each tree. An implementation was obtained from `http://sourceforge.net/projects/iforest/`.

Isolation Forest is parameter-free. For SCiForest, we chose the number of data points to subsample to be 0.66 of available data points and the number of features to consider to be 0.66 of the available features.

4.5 Ensemble Gaussian Mixture Model (egmm)

A classic approach to anomaly detection is to fit a probabilistic model to the available data to estimate the density $P(x)$ of each data point x. Data points of low density are declared to be anomalies. One approach to density estimation is to fit a Gaussian mixture model (GMM) using the EM algorithm. However, a single GMM is not very robust, and it requires specifying the number of Gaussians k. To improve robustness, we generate a diverse set of models by varying the number of clusters k, the EM initializations, and training on 15 bootstrap replicates of the data [29]. We choose a set of possible values for k, $\{6, 7, 8, 9, 10\}$, and try all values in this set. The average out-of-bag log likelihood for each value of k is computed, and values of k whose average is less than 85% of the best observed value are discarded. Finally, each data point x is ranked according to the average log likelihood assigned by the remaining GMMs (equivalent to the geometric mean of the fitted probability densities).

5. SUMMARY OF RESULTS

To assess performance, we employed the AUC (area under the ROC curve). Table 1 provides an overall summary of the algorithms. It shows the number of data sets in which each algorithm appeared in the top 3 algorithms when ranked by AUC (averaged over all settings of difficulty, relative frequency, and clusteredness). We see that Isolation Forest (IF) is the top performer, followed by EGMM and SCIF.

Table 1: # Times in Top 3

egmm	if	lof	ocsvm	scif	svdd
14	**17**	6	5	13	2

To quantify the impact of each of the design properties (relative frequency, point difficulty, and clusteredness) as well as the relative effect of each algorithm and "mother" data set, we performed an ordinary linear regression to model the *logit(AUC)* of each replicate data set as a linear function of each of these factors. The logit transform ($\log[AUC/(1 - AUC)]$) transforms the AUC (which can be viewed as a probability) onto the real-valued log-odds scale. We employed the following R formula:

$$logit(AUC) \sim set + algo + diff + rfreq + cluster \quad (1)$$

where, *set* is the dataset (*abalone*, *shuttle*, etc.), *algo* is the algorithm, *diff* is the point difficulty level, *rfreq* is the relative frequency of anomalies in the benchmark, and *cluster* is the clusteredness of the anomaly class. The *diff*, *rfreq*, and *cluster* values were binned into qualitative factors as described above. Despite the simplicity of this model, inspection of the residuals showed that it gives a reasonable fit.

We found all factors included in the regression to be significant ($p \ll 0.001$, t-test). Figure 1 shows that as the point difficulty increases, the performance degrades for all algorithms. Error bars in this and all subsequent figures show $\pm$ one standard error for the estimates from the regression. Figure 2 shows that anomalies are harder to detect as they become more frequent. And Figure 3 shows that they become harder to detect as they become more clustered. These results all confirm that the benchmark data sets achieve our design goals.

Figure 4 shows the performance on all datasets relative to *abalone*. Anomalies were hardest to detect for *yearp* and easiest for *wave*. Finally, Figure 5 shows the contribution of each algorithm to the logit(AUC) relative to EGMM. This suggests that EGMM and Isolation Forest are giving very similar performance, while the other algorithms are substantially worse.

We also fit a version of Equation (1) with pairwise interaction terms between algorithm, point difficulty, relative frequency, and clusteredness. Very few of these interactions

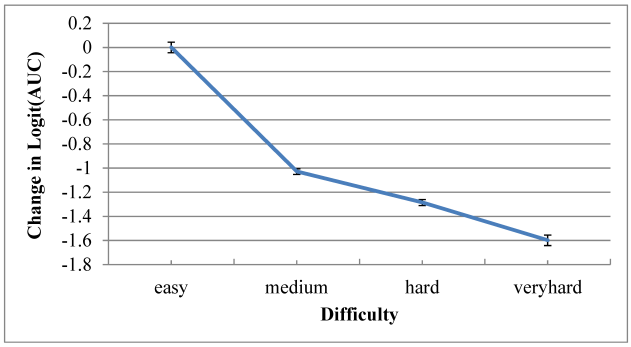

Figure 1: Change in Logit(AUC) with Difficulty

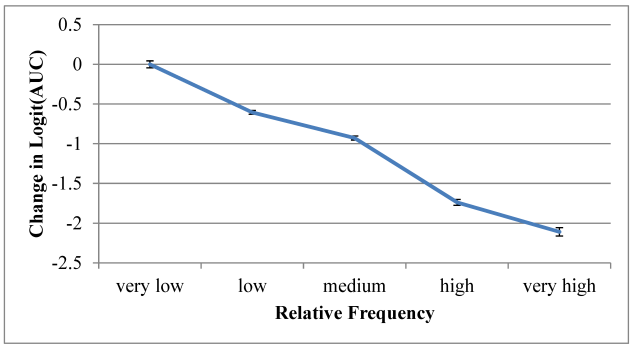

Figure 2: Change in Logit(AUC) with Rel. Freq.

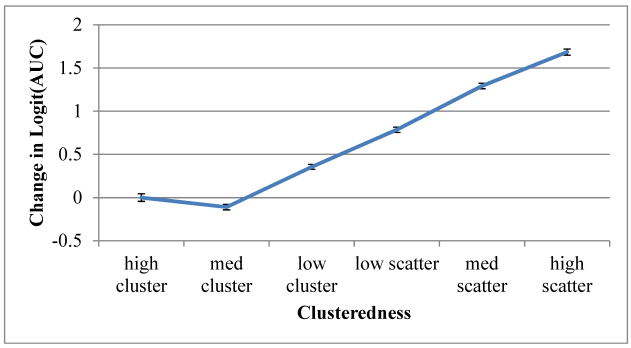

Figure 3: Change in Logit(AUC) with Clusteredness

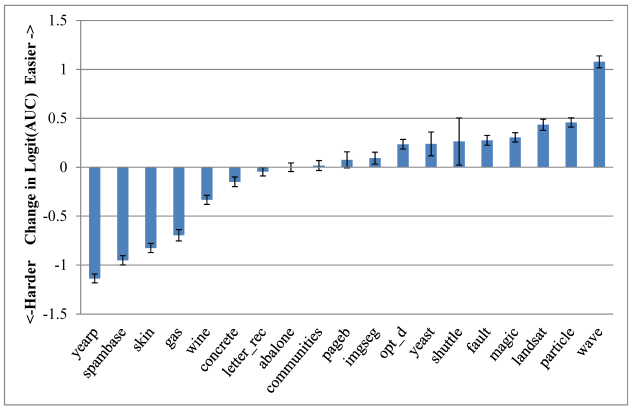

Figure 4: Performance on Datasets

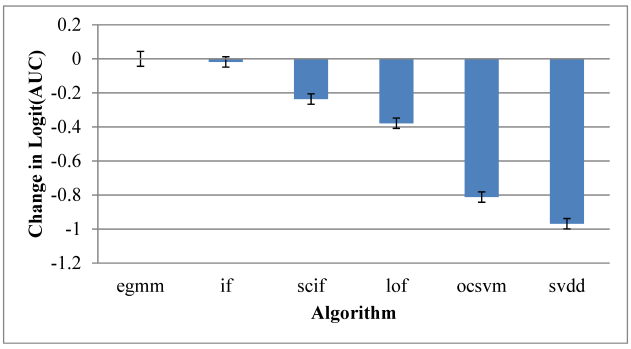

Figure 5: Change in Logit(AUC) with Algorithm

were statistically significant, which confirms that our simple model gives a good characterization of the benchmarks.

6. CONCLUSIONS

We have described a methodology for creating anomaly detection benchmarks and techniques for controlling three important properties of those benchmarks (*point difficulty*, *relative frequency* and *clusteredness*). Experimental tests based on thousands of replicate data sets demonstrate that these three properties strongly influence the behavior of several leading anomaly detection algorithms.

7. FUTURE WORK

We consider these results a work in progress and intend to develop this study further. Our plan is to include more algorithms and metrics and to provide additional statistical analysis of the results. This will include more rigorous statistical justification for our findings, an empirical comparison of algorithms, and an exploration of which settings cause shifts in the relative performance of the algorithms.

An important goal for future work is to validate the predictive value of our benchmarks against real anomaly detection problems. In particular, if we measure the point difficulty, relative frequency, and clusteredness of a real problem, does the most similar benchmark problem predict which anomaly detection algorithms will work best on the real problem? Another important goal is to develop a method for controlling the proportion of relevant (versus irrelevant) features. This would help the research community develop better methods for feature selection in anomaly detection algorithms.

8. ACKNOWLEDGMENTS

Funding was provided by the U.S. Army Research Office (ARO) and Defense Advanced Research Projects Agency (DARPA) under Contract Number W911NF-11-C-0088. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

9. REFERENCES

[1] A. Alzghoul and M. Löfstrand. Increasing availability of industrial systems through data stream mining.

Computers & Industrial Engineering, 60(2):195 – 205, 2011.

[2] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[3] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[4] M. Breunig, H.-P. Kriegel, R. T. Raymond T. Ng, and J. Sander. LOF: identifying density-based local outliers. *ACM SIGMOD Record*, pages 93–104, 2000.

[5] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15:1–15:58, July 2009.

[6] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[7] E. Dereszynski and T. G. Dietterich. Spatiotemporal models for anomaly detection in dynamic environmental monitoring campaigns. *ACM Transactions on Sensor Networks*, 8(1):3:1–3:26, 2011.

[8] J. Glasser and B. Lindauer. Bridging the gap: A pragmatic approach to generating insider threat data. In *2013 IEEE Security and Privacy Workshops*, pages 98–104. IEEE Press, 2013.

[9] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38(0):293 – 306, 1985.

[10] J. Greensmith, J. Twycross, and U. Aickelin. Dendritic cells for anomaly detection. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 664–671. IEEE, 2006.

[11] V. J. Hodge and J. I. M. Austin. A survey of outlier detection methodologies. *AI Review*, 22:85–126, 2004.

[12] T. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proceedings of the 1999 Conference on AI and Statistics*, volume 126, pages 00–04. San Mateo, CA, 1999.

[13] S. Keerthi, K. Duan, S. Shevade, and A. Poo. A fast dual algorithm for kernel logistic regression. *Machine Learning*, 61(1-3):151–165, 2005.

[14] J. S. Kim and C. Scott. Robust kernel density estimation. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 3381–3384, 2008.

[15] T. Lane and C. E. Brodley. Sequence matching and learning in anomaly detection for computer security. In *AAAI Workshop: AI Approaches to Fraud Detection and Risk Management*, pages 43–49, 1997.

[16] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *In Proceedings of SIAM Conference on Data Mining*, 2003.

[17] A. Liu, C. Martin, T. Hetherington, and S. Matzner. A comparison of system call feature representations for insider threat detection. In *Information Assurance Workshop, 2005. IAW '05. Proceedings from the Sixth Annual IEEE SMC*, pages 340–347, 2005.

[18] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *Proceedings of the IEEE International Conference on Data Mining*, pages 413–422, 2008.

[19] F. T. Liu, K. M. Ting, and Z.-H. Zhou. On detecting clustered anomalies using SCiForest. In *Machine Learning and Knowledge Discovery in Databases*, pages 274–290, 2010.

[20] M. Markou and S. Singh. Novelty detection: a review - part 1: statistical approaches. *Signal Processing*, 83(12):2481–2497, 2003.

[21] D. Pokrajac, A. Lazarevic, and L. Latecki. Incremental local outlier detection for data streams. In *Computational Intelligence and Data Mining, 2007. CIDM 2007. IEEE Symposium on*, pages 504–515, 2007.

[22] K. Polat, S. Sahan, H. Kodaz, and S. Günes. A new classification method for breast cancer diagnosis: Feature selection artificial immune recognition system (fs-airs). In L. Wang, K. Chen, and Y. Ong, editors, *Advances in Natural Computation*, volume 3611 of *Lecture Notes in Computer Science*, pages 830–838. Springer Berlin Heidelberg, 2005.

[23] L. Portnoy, E. Eskin, and S. Stolfo. Intrusion detection with unlabeled data using clustering. In *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001*. Citeseer, 2001.

[24] B. Schölkopf, J. C. Platt, J. Shawe-taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution, 1999.

[25] Tax and Duin. Support vector data description. *Machine Learning*, 54:45–66, 2004.

[26] K. L. Wagstaff, N. L. Lanza, D. R. Thompson, T. G. Dietterich, and M. S. Gilmore. Guiding scientific discovery with explanations using DEMUD. In *Proceedings of the Association for the Advancement of Artificial Intelligence AAAI 2013 Conference*, 2013.

[27] F. Xue, W. Yan, N. Roddy, and A. Varma. Operational data based anomaly detection for locomotive diagnostics. In *International Conference on Machine Learning*, pages 236–241, 2006.

[28] B. Zhang, C. Sconyers, C. Byington, R. Patrick, M. Orchard, and G. Vachtsevanos. Anomaly detection: A robust approach to detection of unanticipated faults. In *Prognostics and Health Management, 2008. PHM 2008. International Conference on*, pages 1–8, 2008.

[29] Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman and Hall/CRC, 2012.

[30] J. Zhu and T. Hastie. Kernel logistic regression and the import vector machine. In *Journal of Computational and Graphical Statistics*, pages 1081–1088. MIT Press, 2001.

[31] A. Zimek, E. Schubert, and H.-P. Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining*, 5(5):363–387, 2012.