Anomaly Detection by Combining Decision Trees and Parametric Densities

Matthias Reif, Markus Goldstein, Armin Stahl
German Research Center for Artificial Intelligence (DFKI)
{*reif,goldstein,stahl*}*@iupr.dfki.de*

Thomas M. Breuel
Technical University of Kaiserslautern, Department of Computer Science
breuel@dfki.uni-kl.de

Abstract

In this paper a modified decision tree algorithm for anomaly detection is presented. During the tree building process, densities for the outlier class are used directly in the split point determination algorithm. No artificial counter-examples have to be sampled from the unknown class, which yields to more precise decision boundaries and a deterministic classification result. Furthermore, the prior of the outlier class can be used to adjust the sensitivity of the anomaly detector. The proposed method combines the advantages of classification trees with the benefit of a more accurate representation of the outliers. For evaluation, we compare our approach with other state-of-the-art anomaly detection algorithms on four standard data sets including the KDD-Cup 99. The results show that the proposed method performs as well as more complex approaches and is even superior on three out of four data sets.

1 Introduction

Anomaly and outlier detection have become an important task in pattern recognition for a variety of applications like network intrusion detection, fraud detection, medical monitoring or manufacturing. Outliers are data records which are very different from the normal data but occur only rarely. If they are completely missing within the training data the problem is also called one-class classification or novelty detection.

Many different approaches have been proposed for solving the outlier detection problem: statistical methods, distance and model based approaches as well as profiling methods. Surveys of the diverse categories and methodologies are given in [5], [8] and [9].

Generating artificial counter-examples from the unknown class is also used for outlier detection [1][2][4]. The advantage of this method is that a standard generic classifier can be trained to separate the regular class (or classes) and the anomalous class. These methods perform very well, but the sampling of the anomalous class can be a tricky task, especially when dealing with high dimensionality. Abe et al. [1] apply Active Learning and Lazarevic et al. [7] apply Feature Bagging to address this problem using a decision tree as classifier.

In this paper we propose an extension for standard decision tree algorithms like C4.5 [10] or CART [3] for anomaly detection that can deal with continuous as well as with symbolic features. With the presented method, there is no need of generating artificial samples of the missing class into the training set. Instead, it uses a parametric distribution of the outlier class during the determination of the split points. This avoids problems like the trade-off between the precision of sampling and the prior of the classes. Furthermore, split points are more accurate and training is faster due to fewer samples.

1.1 Decision Trees

Decision trees have several advantages compared to other classification methods, which make them more suitable for outlier detection. In particular they have an easily interpretable structure and they are also less susceptible to the curse of dimensionality [5].

In a decision tree, every node divides the feature space from its parent node into two or more disjoint subspaces and the root node splits the complete feature space. The tree building process selects at each node that split point, which divides the given subspace and the training data best according to some impurity measure. One example impurity measure of a node t and its range in feature space over a set of classes C uses the number of instances of a class $N_c(t)$ and the total number of instances $N(t)$ at node t in the training data [3]:

$$i(t) = -\sum_{c \in C} \frac{N_c(t)}{N(t)} \log_2 \left(\frac{N_c(t)}{N(t)} \right) \qquad (1)$$

The best split at a node is defined as the split with the highest decrease of impurity of its child nodes [3]. The decrease of impurity of a split s at node t with the child nodes t_L and t_R is calculated for binary trees as follows:

$$\Delta i(s,t) = i(t) - \frac{N(t_L)}{N(t)} i(t_L) - \frac{N(t_R)}{N(t)} i(t_R) \quad (2)$$

Since we avoid the sampling of the outlier class c_A, we have to use a density distribution for estimating N_{c_A} instead. In general, it is possible to use any distribution, e.g. if there is some background knowledge available like dependencies of features. In the following, we use for the sake of generality a uniform distribution.

2 Uniform Outlier Distributions

In this section we introduce two uniform distributions used for the outlier class density estimation for the two types of attributes: discrete and continuous.

2.1 Discrete Uniform Distribution

A discrete uniform distribution is defined over a finite set S of possible values and all of these values are equally probable. If there are $|S|$ possible values, the probability of a single one being picked is $\frac{1}{|S|}$. Hence, the probability that a feature has a value out of a set $M \subset S$ is:

$$P(X \in M) = \frac{|M|}{|S|} \quad (3)$$

This requires that the amount of possible symbols is known at the training. Typically it can be derived directly from the training set since each symbol occurs at least once in it. However, if $|S|$ is undersized (a new value of a discrete feature occurs for the first time at classification) and consequently $P(X \in S) < 1$, the according data point can be classified as novel with a high confidence. In this case the probabilities were not estimated completely correct, but with a sufficient accuracy for our task.

2.2 Continuous Uniform Distribution

A continuous uniform distribution defines a constant probability density over a finite interval $[r^{min}, r^{max}]$ with $r^{min} < r^{max}$:

$$f(x) = \begin{cases} \frac{1}{r^{max} - r^{min}} & x \in [r^{min}, r^{max}] \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Since decision trees use intervals within the continuous feature spaces, we are interested in the probability that a data point is located inside such a specific interval $[a,b] \subset [r^{min}, r^{max}]$:

$$P(X \in [a,b]) = \int_a^b f(x)dx = \frac{b-a}{r^{max} - r^{min}} \quad (5)$$

Like the number of symbols $|S|$ for symbolic features, r^{min} and r^{max} have to be defined for all continuous attributes separately before the training such that $P(X \in [r^{min}, r^{max}]) = 1$. In general it is a good idea that r^{min} and r^{max} do not coincide with the smallest and largest value of the training set since we want the uniform distribution to have a "border" around the data in order to find optimal split points. There are many different approaches conceivably for defining the border values like adding (subtracting respectively) a fixed rate to the biggest/smallest value occurring in the training set [11], e.g. 10% of the difference between both. A more convincing method is to use a value, that is derived from the actual distribution of the given data, like a multiple of the variance.

2.3 Joint Distribution

For testing a split, a decision tree algorithm needs to know the number of instances which fall into the resulting multidimensional subspaces. In our approach, the number for the class representing the anomaly is determined based on a uniform probability distribution and not on previously generated data points.

We can compute the probability that a data point of a uniform distribution over the limited space lies inside such a subspace Q, that is defined by its intervals $[a_i, b_i] \subset [r_i^{min}, r_i^{max}]$ for all k_c continuous features and its subsets $M_j \subset S_j$ of all k_s symbolic features. Under the assumption that the features are independent variables this is the joint probability of its single probabilities from Equation (3) and (5):

$$P(X \in Q) = \prod_{i=1}^{k_c} P(X_i \in [a_i, b_i]) \prod_{j=k_c+1}^{k_c+k_s} P(X_j \in M_j) \quad (6)$$

Having N_n regular training samples in total, we can calculate the expected number of instances $N_{c_A}(t)$ of the anomaly class c_A within the subspace Q_t of node t to use it in Equation (1) by using their prior probability $P(c_A)$:

$$N_{c_A}(t) = N_{c_A} P(X \in Q_t) = \frac{P(c_A)}{1 - P(c_A)} N_n P(X \in Q_t) \quad (7)$$

The prior of the anomalous class $P(c_A)$ mainly controls the trade-off between detection rate (correctly classifying outliers as novel) and false alarm rate (wrongly classifying normal instances as novel). Varying this parameter results in the ROC curve as illustrated in Figure 2.

3 Methodology

Wherever in the decision tree algorithm the number of instances of the unknown class is needed, we can

now use Equation (7) to estimate it without the need for artificially generated samples. There is also no need for major changes in the procedure of finding the next best split, which makes the proposed method applicable to arbitrary decision tree algorithms. Efficient techniques to calculate the quality of a split as proposed in [12] are also useable. However, there are several issues when dealing with the proposed method, which we consider in the following subsections.

3.1 Suitable Split Points

A decision tree algorithm usually checks a lot of potential split points in order to find the one, which divides the training data best. For symbolic features we consider symbol sets without any order. The splits are only checked on the basis of single symbols. Under the assumption that all symbols are already known during the tree building procedure, there is no reason to consider other split points in our proposed method than in other standard decision tree creation procedures.

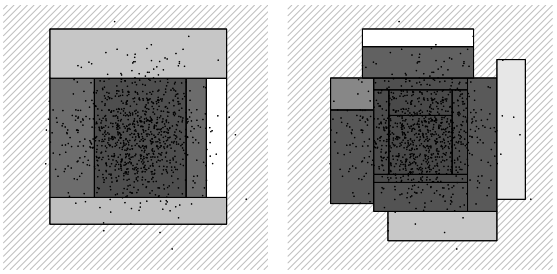
For continuous features, the determination of good potential split points is more difficult. The standard way is to use the mean of two successive feature values that occurred in the training set as candidate split points. Since in our modified tree we only have instances of one, respectively not every class, this would result in splits between data points of the regular classes only. This is a crucial point for delimiting regular classes from areas without training samples.

A possible solution is using a grid over the feature space such that splits will be tested in constant intervals. This brute force method has the drawback, that the search will spend a lot of time testing splits in areas of the feature space, where known data points (and therefore optimal splits) are rare.

Therefore we test split points close to given samples with a defined spacing. The spacing distance should be chosen by considering the data. We compute it by the mean of all distances between successive data points. It is conceivable that it could also be determined by the mean and variance of all data points, but our experiments show that this definition plays a minor role. Further on, it is also possible to create arbitrary more potential split points around a known data point with fractions of the distance, recalling that the tree will select the best cut based on the minimal impurity anyway.

3.2 Stopping Criterion

A standard tree building algorithm stops the creation of new child nodes if a predefined stopping criterion is fulfilled, e.g. a lower error bound is reached. A special case is setting this error bound to zero such that all training samples are classified correctly. This is not possible when dealing with outlier distributions, because there will always be a misclassification error at a leaf node (within computational precision). Therefore we have to define an error limit $\epsilon > 0$ as threshold. It should be selected with respect to $P(c_A)$ in relation to the dimensionality of the data. If $P(c_A)$ is quite small and the dimensionality is rather high, a smaller value is possibly required to force the algorithm to cut closer to the data points in order to increase the detection rate.



(a) lower $P(c_A)$ (b) higher $P(c_A)$

Figure 1. Different decision boundaries on synthetic 2D example by varying the outlier prior

3.3 Pruning

A nice property of decision trees is the possibility to prune them in order to increase their generalization performance. There are several techniques like leave-one-out or n-fold cross validation, which are still applicable with our approach since they simply use error measures based on counting instances. But the effect of pruning is lower when using the proposed extension because dividing the outlier class has no influence at all. Our evaluation shows that the method obtains good results even without pruning.

4 Experiments and Evaluation

For a first illustrating experiment we created a small synthetic data set that contains 1000 normally distributed data points in a two-dimensional space. The visualization of the results for two different values of the prior $P(c_A)$ are shown in Figure 1. Each filled rectangle represents a leaf of the decision tree that classifies its region as the regular class. The brightness indicates the confidence and leaves representing the outliers were grouped together to one hatched area.

For comparing our approach, we performed an evaluation with different public and commonly used real life data sets from the UCI repository. Because they are not designed for novelty detection, we also applied the modifications as described in [1]: The most common class of each data set was chosen to be the known

Dataset	Regular Class	Anomaly Class	Active-Outlier	Bagging	Feature Bagging	Boosting	LOF	This Paper
Ann-thyroid	3	1	0.97	0.98	0.869	0.64	0.869	**0.993**
Ann-thyroid	3	2	0.89	0.96	0.769	0.54	0.761	**0.977**
Shuttle (avg.)	1	2,3,5,6,7	**0.999**	0.985	0.839	0.784	0.825	0.994
KDD-Cup 99	normal	U2R	0.935	0.611	0.74	0.510	0.61	**0.946**

Table 1. AUC for real life data sets after converting them into binary problems

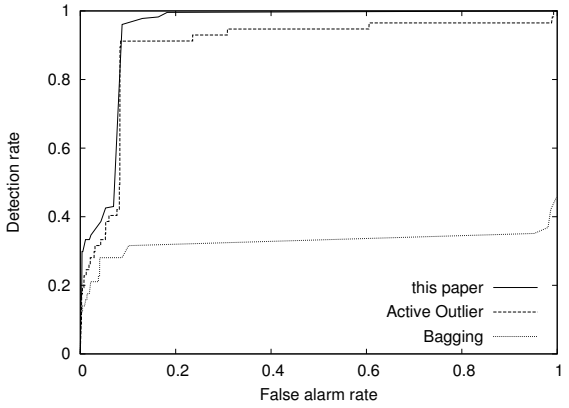


Figure 2. ROC curves for Active Outlier, Bagging and this paper (KDD-Cup 99)

one. The training sets consist of these classes only. The test set contains additionally one of the rare classes to represent anomalies.

To compare the results of the presented approach to other methods [1][6][7], the common concepts of receiver operating characteristic (ROC) and especially the area under the ROC curve (AUC) are used. Table 1 gives an overview of the achieved AUC values for different data sets. In Figure 2 the ROC curves of different approaches for the KDD-Cup 99 data are compared to our method, whereas the most unfavorable results (due to the nondeterministic sampling) have been used following [1].

Summarizing the results, the proposed method performes best on three out of four data sets. The results are slightly better than "Active-Outlier", but the presented approach is appealing by its simplicity, its deterministic results and –depending on the number of suitable split points used– its faster algorithm.

5 Conclusion

In this paper we introduced a new deterministic approach to use decision trees for anomaly detection with no need to generate artificial counter-examples of the outlier class. The general integration of the outlier densities into the tree building process makes it possible to use different decision tree algorithms. The results on multiple data sets showed that our approach is practically as good as state-of-the-art methods or even better.

Acknowledgment

This work is part of NetCentric Security, a project of Deutsche Telekom Laboratories supported by German Research Center for Artificial Intelligence DFKI GmbH.

References

[1] N. Abe, B. Zadrozny, and J. Langford. Outlier detection by active learning. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, 2006. ACM Press.

[2] A. Bánhalmi, A. Kocsor, and R. Busa-Fekete. Counter-example generation-based one-class classification. *Machine Learning: ECML 2007*, 2007.

[3] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Chapman & Hall/CRC, 1984.

[4] W. Fan, M. Miller, S. J. Stolfo, W. Lee, and P. K. Chan. Using artificial anomalies to detect unknown and known network intrusions. In *ICDM*, 2001.

[5] V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artif. Intell. Rev.*, 22(2), 2004.

[6] E. M. Knorr and R. T. Ng. Algorithms for mining distance-based outliers in large datasets. In *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, 1998.

[7] A. Lazarevic and V. Kumar. Feature bagging for outlier detection. In *KDD*. ACM, 2005.

[8] M. Markou and S. Singh. Novelty detection: A review - part 1: Statistical approaches. *Signal Processing*, 83(12), 2003.

[9] M. Markou and S. Singh. Novelty detection: A review - part 2: Neural network based approaches. *Signal Processing*, 83(12), 2003.

[10] R. J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[11] G. Schmidberger and E. Frank. Unsupervised discretization using tree-based density estimation. In *PKDD*, volume 3721 of *Lecture Notes in Computer Science*. Springer, 2005.

[12] J. C. Shafer, R. Agrawal, and M. Mehta. Sprint: A scalable parallel classifier for data mining. In *Proc. 22nd Int. Conf. Very Large Databases, VLDB*. Morgan Kaufmann, 1996.