Avoiding Anomalies in Data Stream Learning

João Gama[1,2], Petr Kosina[1], and Ezilda Almeida[1]

[1] LIAAD-INESC TEC, University of Porto
{ezildacv, petr.kosina}@gmail.com,
[2] Faculty of Economics, University Porto
jgama@fep.up.pt

Abstract. The presence of anomalies in data compromises data quality and can reduce the effectiveness of learning algorithms. Standard data mining methodologies refer to data cleaning as a pre-processing before the learning task. The problem of data cleaning is exacerbated when learning in the computational model of data streams. In this paper we present a streaming algorithm for learning classification rules able to detect contextual anomalies in the data. Contextual anomalies are surprising attribute values in the context defined by the conditional part of the rule. For each example we compute the degree of anomaliness based on the probability of the attribute-values given the conditional part of the rule covering the example. The examples with high degree of anomaliness are signaled to the user and not used to train the classifier. The experimental evaluation in real-world data sets shows the ability to discover anomalous examples in the data. The main advantage of the proposed method is the ability to inform the context and explain why the anomaly occurs.

Keywords: Data Streams, Rule Learning, Anomaly Detection

1 Motivation

The amount of digital data currently handled is huge. Our ability to collect huge amounts of detailed information is increasing exponentially. Nevertheless, data anomalies such as inconsistencies, missing values, outliers, etc. are more frequent than desired. The existence of these data problems, commonly called *dirty data*, degrades the quality of the information with direct impact on the efficiency of data analysis techniques [1]. These problems can lead to incorrect decisions or strategies that often are costly to organizations. Improving data quality by detecting and eliminating errors and inconsistencies in the data is a relevant data mining problem.

Standard data mining methodologies define *pre-processing* as a key step before the learning phase. Pre-processing is essential to analyze the multivariate data sets before data mining. In this step, the data set is cleaned, by removing observations containing noise and those with missing data. The identification of observations that are not coherent with the rest of the data, can be used in two different perspectives. One perspective consists of removing these observations from the analysis. The other perspective considers these observations as interesting and therefore is very important to detect. Depending on the application, the user can be especially interested in the anomalous cases more than in the 'normal' observations. They may represent malicious cases such as intrusions, frauds or diseases. One way or another, the detection of anomalies is very important.

Most of the works in pre-processing data, anomaly and outlier detection are off-line. Current tasks usually consist of many observations being processed and the number is still increasing. It is increasing to the extent that the relatively recent approach called stream mining considers the data possibly infinite. Finding anomalies in such a setting is especially a difficult task not only because of the potentially unbounded size, but also because one of the typical characteristics of data stream is that the distribution generating the data can change over time.

This work presents a method for on-line anomaly detection, a crucial task in real-world applications. The method is embedded in a streaming classification rule learner, although it can be integrated with any VFDT like algorithm. The anomalies detected are characterized by a *context* that refers the region of the instance where the anomaly was detected, and behavioral attributes, those with anomalous values. This is a key advantage of the proposed system: it explains *where* and *why* a given example is anomalous.

The paper is organized as follows. The next section presents the related work in outlier and anomaly detection. Section 3 describes the method used to detect contextual anomalies that has been implemented inside a classification rule learner for data streams. Section 4 describes the anomalies detected in several well-known datasets. The last Section presents the conclusions and futures work.

2 Related Work

Anomaly detection refers to detecting observations that do not conform to an established normal behavior. Anomalies are also referred to as outliers, change, deviation, surprise, aberrant, peculiarity, intrusion, etc [2]. [9] points out the importance of data cleaning in developing real world applications. Data cleaning deals with missing values, noisy data, inconsistent data, etc. In this work, we focus in a particular form of inconsistent data: anomalies or outliers.

Statistical approaches were the earliest algorithms used for outlier detection. The most common approaches are univariate. Probably one of the simplest statistical outlier detection techniques use informal box plots [18] to pushup outliers in both univariate and multivariate data sets. Another single dimensional method was presented in [18] which calculates a Z value as the difference between the mean value for the attribute and the query value divided by the standard deviation for the attribute. The Z value for the query is compared with a 1% or 5% significance level. The technique requires no user parameters as all parameters are derived directly from data.

The literature in anomaly and outlier detection is huge. Two recent overviews, with excellent references are [11] and [2]. Most of the works refer to off-line approaches. A recent paper [15], addresses the anomaly detection problem in large-scale data mining applications using residual subspace analysis. The authors suggest a framework wherein random projection can be used to obtain compressed data. Their contribution shows that the spectral property of the compressed data is approximately preserved under such projection and thus the performance of spectral-based methods for anomaly detection is almost equivalent to the case in which the raw data is completely available.

[19] present an approach for combining adaptive pre-processing with adaptive online predictor. The authors present a case study with real sensory data from a production process. In that case, decoupling the adaptively of pre-processing and the predictor contributes to improving the prediction accuracy.

The authors of [2] define 2 types of anomalies.

- Point Anomalies. If an individual data instance can be considered as anomalous with respect to the rest of data, then the instance is termed as a point anomaly. This is the simplest type of anomaly and is the focus of majority of research on anomaly detection.
- Contextual Anomalies. If a data instance is anomalous in a specific context.. In this case, it is convenient to define:
  - Contextual attributes. The contextual attributes are used to determine the context for that instance.
  - Behavioral attributes. The attributes with abnormal values in the contexts defined by the contextual attributes.

A relevant aspect, pointed out by [2], is that an observation might be an anomaly in a given context, but an identical data instance (in terms of behavioral attributes) could be considered normal in a different context. This property is a key characteristic in identifying contextual and behavioral attributes for a contextual anomaly detection technique.

In [13], the authors discuss distance-based outlier detection methods for very large data bases, and propose several algorithms. The most efficient has complexity that is linear with the number of examples but exponential in the number of attributes. It is based on nearest neighbour search over cells defined by indexing structures. While the proposed algorithms are effective for very large data bases, its complexity limit their applicability in the streaming computational model.

One of the few systems that can detect contextual anomalies is Gritbot [16]. GritBot is not described in any scientific paper, is a commercial tool that detects inconsistencies in the data set. GritBot is an off-line tool that finds anomalies in data as a pre-processing to data mining algorithms. It can be thought as an autonomous data quality auditor that hunts for records having ”surprising” values of nominal and/or numeric attributes. Anomalies need not stand out in the complete dataset – GritBot searches for subsets of records in which the anomaly is apparent. Although there is no technical description of the methods used by GritBot, we can guess from the code, results and studies published by Quinlan that the GriBot generates rules iteratively. In each iteration, considers an attribute from a subset of n attributes as objective (dependent) attribute. Then, to each tuple that violates a certain rule is assigned the probability that the value anomaly can occur by chance and not by error. The approach we present in this paper identifies anomalies *a la* Gritbot, but on-line, with a single-scan over the data.

3 Anomaly Detection

The method we propose detects contextual anomalies. Contextual anomalies are characterized by a *context* that refers the region of the instance space where the anomaly

was detected, and behavioral attributes, those with anomalous values. One example of the type of anomalies we detect, from the Adult dataset [5], is:
Case 15904:
```
education = 10th [6 in 1470]
capital-gain = 99999 [889.2±633.6]
```
Rule:
```
education-num <= 10 ∧
marital-status = Married-civ-spouse → >50K
```

The 15904th example is signaled as an anomaly, and is interpreted as follows. The context of the anomaly is given by the rule:
```
education-num <= 10 ∧
marital-status = Married-civ-spouse → >50K.
```
The attributes with suspicious values are *education* and *capital-gain*. The first attribute is nominal. In 1470 examples, the attribute value *education = 10* was observed 6 times. The second attribute is numerical. The mean of this variable (using the examples seen so far) is 889 and the standard deviation is 633. The anomaliness score for this example is 0.99.

In the first part of this section, we describe the algorithm to learn the decision rules defining the context of the anomalies. We should point out, that our anomaly detection system can be used in classification and regression problems with any VFDT like algorithms [4]. The current implementation is based on a stream classification rule learner, previously presented in [8]. In the next Section we provide a concise description of the learning algorithm, to clarify how the detection method works.

3.1 Very Fast Decision Rules Algorithm

As in many other systems, a rule in VFDR [14] is an implication of the form $A \Rightarrow C$. The A part of a rule is a conjunction of literals, that is, conditions based on attribute values. For numerical attributes, each literal is of the form $X_i > v$, or $X_i \leq v$ for some feature X_i and some constant v. For categorical attributes VFDR produce literals of the form $X_i = v_j$ where v_j is a value in the domain of X_i. The C part of a rule r, designated $\mathcal{L}_r$, is not a constant as in most of rule based systems, but a function. This is the most different feature of VFDR.

The VFDR algorithm is designed for high-speed data streams. It learns ordered or unordered rule sets. It needs only one scan of data and is able to provide any-time classifications.

Growing a Set of Rules The algorithm begins with a empty rule set (RS) and a $default\ rule$ $\{\} \rightarrow \mathcal{L}$, where $\mathcal{L}$ is initialized to $\emptyset$. $\mathcal{L}$ is a data structure that contains information used to classify test instances, and the sufficient statistics needed to expand the rule.

As already said, each learned rule (r) is a conjunction of literals, that are conditions based on attribute values, and a $\mathcal{L}_r$. If all the literals are true for a given example, then the example is said to be *covered* by the rule. The labeled examples covered by a rule

r are used to update $\mathcal{L}_r$. A rule is expanded with the literal that has the highest gain measure of the examples covered by the rule. $\mathcal{L}_r$ accumulates the sufficient statistics to compute the gain measure of all possible literals. $\mathcal{L}_r$ is a data structure that contains: an integer that stores the number of examples covered by the rule; a vector to compute $p(c_k)$, i.e., the probability of observing examples of class c_k; a matrix $p(X_i = v_j|c_k)$ to compute the probability of observing value v_j of a nominal attribute X_i per class; and a `btree` to compute the probability of observing values greater than v_j of continuous attribute X_i, $p(X_i > v_j|c_k)$, per class. The information maintained in $\mathcal{L}_r$ is similar to the sufficient statistics [6].

The number of observations, after which a rule can be expanded or new rule can be induced, is determined by the Hoeffding bound. It guarantees that, with probability at least $1 - \delta$, the true mean of a random variable x with a range R will not differ from the sample mean of size N by more than:

$$\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2N}}.$$

It is not efficient to check for the sufficient number of examples with every incoming example, therefore this is done only after every N_{min} observations.

The set of rules (RS) is learned in parallel as described in Algorithm 1. We consider two cases: learning ordered or unordered set of rules. In the former, every labeled example updates statistics of the first rule that covers it. In the latter, every labeled example updates statistics of all the rules that cover it. If a labeled example is not covered by any rule, the $default\ rule$ is updated.

The expansion of a rule is done using Algorithm 2 that employs the aforementioned Hoeffding bound. For each attribute X_i the value of split evaluation function G is computed for each attribute value v_j. If the best merit is better the second best with given confidence, i.e. satisfies condition $g_{best} - g_{2best} > \epsilon$, the rule is expanded with condition $X_a = v_j$ and the class of the rule is assigned according to the majority class of observations of $X_a = v_j$.

Classification Strategies The set of rules learned by `VFDR` can employ different classification strategies: *First Hit*, and *Weighted Sum*. As in [3], the ordered rules use the *First Hit* strategy, while the unordered rules use the *Weighted Sum* strategy. In that case all rules covering the example are used for classification and the final class is decided by using weighted vote.

More specifically, assume that a rule r covers a test example. The example will be classified using the information in $\mathcal{L}_r$ of that rule. The simplest strategy uses the distribution of the classes stored in $\mathcal{L}_r$, and classify the example in the class that maximizes $p(c_k)$. This strategy only use the information about class distributions and does not look for the attribute-values, therefore it uses only a small part of the available information. In a more informed strategy, a test example is classified with the class that maximizes the posteriori probability given by Bayes rule assuming the independence of the attributes given the class. There is a simple motivation for this option. $\mathcal{L}$ stores information about the distribution of the attributes given the class usually for hundreds

Algorithm 1: VFDR: Rule Learning Algorithm.

```
input  : $S$: Stream of examples
         $N_{min}$: Minimum number of examples
         $ordered\_set$: boolean flag
output: $RS$: Set of Decision Rules
begin
    Let $RS \leftarrow \{\}$
    Let $default\ rule\ \mathcal{L} \leftarrow \emptyset$
    foreach example $(x, y_k) \in S$ do
        foreach Rule $r \in RS$ do
            if $r$ covers the example then
                Update sufficient statistics of Rule $r$
                if Number of examples in $\mathcal{L}_r$ mod $N_{min} = 0$ then
                    $r \leftarrow ExpandRule(r)$
                if $ordered\_set$ then
                    BREAK
        if none of the rules in $RS$ trigger then
            Update sufficient statistics of the empty rule
            if Number of examples in $\mathcal{L}$ mod $N_{min} = 0$ then
                $RS \leftarrow RS \cup$ ExpandRule($default\ rule$)
```

or even thousands of examples, before expanding the rule and re-initializing the counters. Naive Bayes (*NB*) takes into account not only the prior distribution of the classes, but also the conditional probabilities of the attribute-values given the class. This way, there is a much better exploitation of the available information in each rule. Given the example $\boldsymbol{x} = (x_1, \ldots, x_j)$ and applying Bayes theorem, we obtain:

$$P(c_k|\boldsymbol{x}) \propto P(c_k) \prod P(x_j|c_k).$$

Using *NB* in VFDT like algorithms [4], is a well-known technique since it was introduced in [7]. One of its greatest advantages is the boost in any-time learning property because even though the learned rule set might not be robust enough or the individual rules might not provide sufficient information for expert interpretation (not being specialized enough, i.e., having only one or few conditions), it may already be able highly informed predictions based on *NB* classification.

3.2 Detecting Anomalies

Different kinds of rule systems are commonly used in multivariate anomaly detection. The use of AVFDR in on-line detection is one of the advantages the system provides. It can detect possible anomalies during the learning process. The detection process works as follows. When the system reads a new example, the rule set is checked to find the rules that cover the example. An example is covered by a rule, when the conditional

Algorithm 2: `ExpandRule`: Expanding one Rule.

```
input  : $r$: One Rule
         $G$: Split evaluation function;
         $\delta$: is one minus the desired probability
         of choosing the correct attribute;
output: $r$: Expanded Rule
begin
    Compute $\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2N}}$ (Hoeffding bound)
    $EvaluateLiterals()$
    if $(g_{best} - g_{2best} > \epsilon)$ then
        Extend $r$ with a new condition based on the best attribute $X_a = v_j$
        Release sufficient statistics of $\mathcal{L}_r$
        $r \leftarrow r \cup \{X_a = v_j\}$
    return $r$
```

tests of the antecedent of the rule are true for that example. For each attribute value, we compute the probability $P(X_i = v|Rule_r)$. These probabilities are computed from the consequent of the rule, $\mathcal{L}_r$, that maintains the sufficient statistics required to expand the rule. Low values of these probabilities suggest that the example is an uncommon case in the context of the rule, and it is reported as anomaly. More specifically, for an example $(\boldsymbol{x}, y)$ and its attribute $X_i = v$ let

$$\Pr(X_i = v|\mathcal{L}_r)$$

be the probability of observing attribute value v of the attribute X_i given the conditions of a rule r.

We compute the univariate anomaliness score as:

$$Uscore_i = 1 - \Pr(X_i = v|\mathcal{L}_r) \tag{1}$$

This score is unsupervised in the sense that does not take into account the class label of the example. During the on-line learning, the learner receives labeled examples and therefore we can use the class information to compute the univariate score. The supervised univariate anomaly score is given by:

$$U^s score_i = 1 - \Pr(X_i = v|y, \mathcal{L}_r) \tag{2}$$

If $Uscore_i > \lambda$, for a given value of λ (typically 99%), the attribute value is said to be an anomaly for the context provided by rule r. This is applicable both for supervised and unsupervised scores.

Computing the Anomaly Score for Nominal Attributes The domain of nominal is finite and unordered. For each nominal attribute, the statistics store in $\mathcal{L}_r$ of a rule, are in the form of a contingency table. For a given attribute, let N be the number of examples, seen so far, covered by rule r, $N_{i,\cdot}$ the number of examples, covered by rule

r, where the attribute take the i^{th} value, $N_{\cdot,j}$ the number of examples, covered by rule r, from class j, and $N_{i,j}$ be the number of examples where the attribute takes value i in examples of class j.

Assume we observe an example of class c where the value of attribute i is v. The univariate anomaliness score for this attribute is computed as:

$$Uscore_i = 1 - \frac{N_{v,\cdot}}{N} \tag{3}$$

The supervised anomaly score is computed as:

$$U^s score_i = 1 - \frac{N_{v,c}}{N_{\cdot,c}} \tag{4}$$

Again, if $Uscore_i > \lambda$, the attribute value is said to be an anomaly for the context provided by rule r.

Computing the Anomaly Score for Continuous Attributes For continuous attributes, the statistics stored in $\mathcal{L}_r$ include the mean and standard deviation of each attribute given the class. Remember that these statistics are computed from the examples covered by the rule. Using these statistics we can compute Equation 1 (or Equation 2) using different strategies, including Normal distribution, Z scores, etc. From a set of experiments not described here, the Chebyshev inequality seems to be more effective.

The Chebyshev inequality guarantees that in any probability distribution, 'nearly all' values are close to the mean. More precisely no more than $\frac{1}{k^2}$ of the distribution's values can be more than k standard deviations away from the mean. Although conservative, the inequality can be applied to completely arbitrary distributions (unknown except for mean and variance). Let x be a random variable with finite expected value $\overline{x}$ and finite non-zero variance σ^2. Then for any real number $k > 0$,

$$\Pr(|x - \overline{x}| \geq k\sigma) \leq \frac{1}{k^2}.$$

Only the case $k > 1$ provides useful information. When $k < 1$ the right-hand side is greater than one, so the inequality becomes vacuous, as the probability of any event cannot be greater than one. When $k = 1$ it just says the probability is less than or equal to one, which is always true.

Therefore, for $k = \frac{|x-\overline{x}|}{\sigma}$ and $k > 1$, the anomaliness score is:

$$Uscore_i = 1 - \frac{1}{(\frac{|x-\overline{x}|}{\sigma})^2} \tag{5}$$

The anomaliness score in Equation 5 can be computed using supervised or unsupervised information depending on computing $\overline{x}$ and σ conditioned to the class or not.

Relatively new rules, that are rules that have not been trained with many examples, would more often tend to report a training example as anomaly. In order to prevent this situation, only rules that were trained with more than m_{min} examples are used in the anomaly detection.

Multivariate Score For each training example and for each attribute value i, we compute the univariate score, $Uscore_i$ using Equation 1. The join degree of anomaliness, assuming that the attributes are independent, is computed for all the attributes such that $Uscore_i > \lambda$, and is given by:

$$\prod_k Uscore_i$$

where k is the set of anomalous attributes.

By applying logarithms, to avoid numerical instabilities, and normalizing, the degree of anomaliness of an example is given by:

$$Ascore = \frac{\sum_{i=1}^{n} \mathcal{I}(Uscore_i)}{\sum_{i=1}^{n} log(Uscore_i)} \tag{6}$$

where

$$\mathcal{I}(x) = \begin{cases} 0 & \text{if } x < \lambda \\ log(x) & \text{otherwise} \end{cases}$$

Equation 6 takes values in the interval $[0, 1]$, where 0 corresponds to the case that none of the attributes is anomalous, and 1 when all the attributes are anomalous.

3.3 Discussion

The main contribution of this work is the ability of incorporating anomaly detection inside the learning process. The advantages of this integration are two-fold. On one hand, the system reports possibly anomalous values, together with an explanation of why each value seems surprising. This information provides insights to the user about the dynamic of the process generating data. On the other hand, the online identification of anomalous examples prevents us to learn from outliers. This is a crucial task in online learning.

Although we illustrate the usability of anomaly detection coupled with a decision rule learner, the proposed method can be used in classification and regression problems with any VFDT like algorithm. The information required to compute the anomaly score is stored in the consequent of rules and in the leaves of a decision tree. Algorithms like decision trees [6] and regression trees [12] can easily incorporate the techniques presented here. The proposed method does not guarantee to find all the anomalies. Moreover, what is an anomaly might depend in the order that examples arrive. The set of rules learned by AVFDR are stable with respect to the order of examples [8].

4 Experimental Evaluation

4.1 UCI Datasets

In a firs set of experiences, and for sanity check, we run the on-line anomaly detection in two artificial datasets - waveform21 [5] and SEA [17]. In these datasets the algorithm did not find any anomalies, which is the correct behavior. In the SEA dataset, shown in Figure 1, the anomaly score [1] is always around 0.

[1] The y axis in the plots showing the distribution of the anomaliness score is in log scale.

Table 1. Anomaly Detection Summary

	Nr.Anomalies	Prequential error	Error in holdout
Adult			
Normal	-	17.58	17.51
Unsupervised	9	17.57	17.51
Supervised	9	17.57	17.51
Covertype			
Normal	-	24.92	38.46
Unsupervised	57	23.75	32.55
Supervised	37	23.91	36.88
Electricity			
Normal	-	18.77	
Unsupervised	189	18.51	
Supervised	13	18.96	
KDDCup99			
Normal	-	0.86	
Unsupervised	10	0.84	
Supervised	17	0.82	

4.2 Real-World Data

The second set of experiments uses well-known datasets were we find anomalies. A summary of the number of anomalies detected is presented in Table 1. For each dataset, we report 3 lines. The first line reports the behavior of AVFDR without detecting anomalies. The second and third line summarizes the behavior of the system using unsupervised and supervised anomaly detection, respectively. The anomalies detected are not used for training the rule learner. The details about these experiments are reported in the following subsections.

Intrusion Dataset The **KDDCUP 99** is a data set [5] of TCP/IP connections which are labeled either as normal or one of many different types of attacks. In many cases, the attacks are grouped into four categories: DOS (denial-of-service), R2L (unauthorized access from a remote machine), U2R (unauthorized access to local superuser privileges), and probing (surveillance and other probing).

- DOS: denial-of-service, e.g., syn flood;
- R2L: unauthorized access from a remote machine, e.g. guessing password;
- U2R: unauthorized access to local superuser (root) privileges, e.g., various 'buffer overflow' attacks;
- probing: surveillance and other probing, e.g., port scanning.

The test data is not from the same distribution as the training data and moreover there are new attack types that are not in the training data. These new types can be grouped to the categories above as well. The set consist of 4,898,431 and 311,029 instances for training and test respectively.

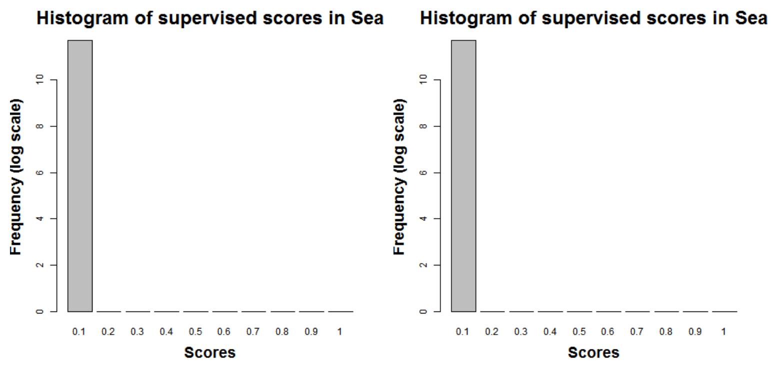

Fig. 1. Distribution of the anomaliness score (supervised and unsupervised) in the SEA dataset.

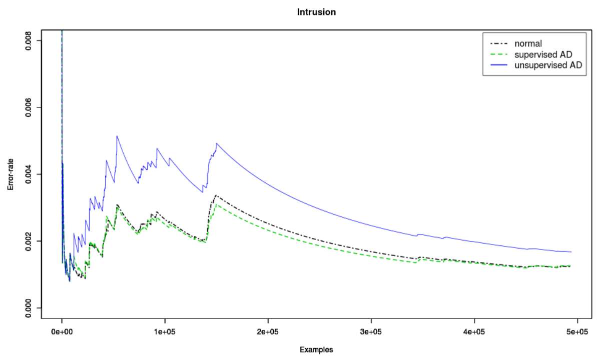

Fig. 2. Anomaly detection influence on prequential error in Intrusion dataset

An example of supervised anomalies found in intrusion dataset:
case 148160:
`dst_host_count=15 [252.2±21.3 class=normal]`
`dst_host_srv_count=13 [248.1±20.9 class=normal]`
in rule:
`count` $\leq 5 \wedge$ `service = private` $\rightarrow$ `Probing`

An example of unsupervised anomalies found in intrusion dataset:
case 121735:
`dst_host_srv_count=163 [254.9±4.51]`
`dst_host_same_srv_rate=0.64 [1.0±0.02]`
in rule:
`count` $> 508 \wedge$ `service = ecr_i` $\rightarrow$ `DoS`

Distribution of the anomaliness score (supervised and unsupervised) in the Intrusion dataset is provided in 3.

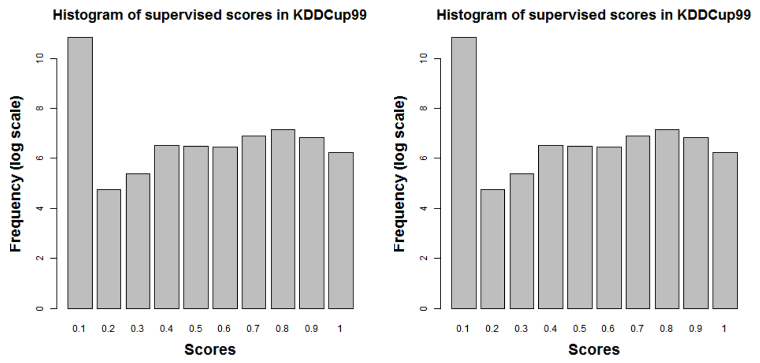

Fig. 3. Distribution of the anomaliness score (supervised and unsupervised) in the Intrusion dataset.

Adult Dataset Data was extracted from the census bureau database in 1994. Prediction task is to determine whether a person makes over 50K a year. The description of data in UCI [5] refers: *A set of reasonably clean records*. The distribution of the anomaliness score (supervised and unsupervised) in the Adult dataset4. Examples of supervised anomalies found in adult dataset:

case 1231

`occupation = Priv-house-serv [0 in 216 class=`$\leq 50K$`]`

`native-country = France [0 in 216 class=`$\leq 50K$`]`

in rule:

`education-num > 12` $\wedge$ `marital-status = Never-married` $\rightarrow$

$\rightarrow \leq 50K$

case 98361:

`occupation = Tech-support [3 in 625 class=`$\leq 50K$`]`

`native-country = Peru [0 in 625 class=`$\leq 50K$`]`

in rule:

`age` > 35 $\wedge$ `education-num` $<= 9$ $\wedge$ `marital-status = Married-civ-spouse`

$\rightarrow > 50K$

Electricity Dataset A widely used dataset is the Electricity Market Dataset introduced in [10]. This time series based data was collected from the Australian New South Wales Electricity Market. The class label identifies the change of the price related to a moving average of the last 24 hours.

Examples of supervised anomalies found in electricity dataset:

case: 7123

`day = 6 [0 in 185 class UP]`

in rule:

`nswprice` > 0.102 $\rightarrow$ `UP`

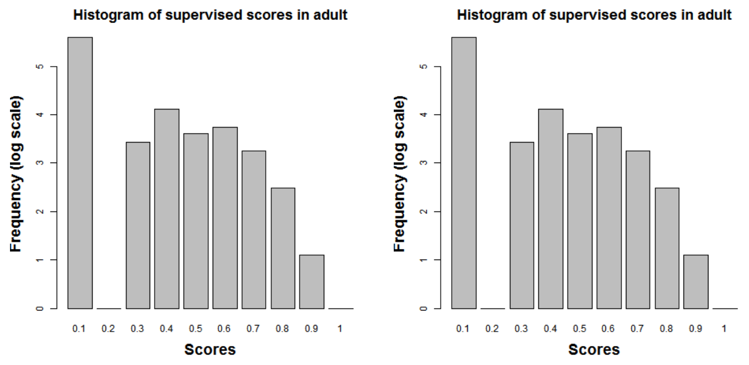

Fig. 4. Distribution of the anomaliness score (supervised and unsupervised) in the Adult dataset.

The distribution of the anomaliness score (supervised and unsupervised) in the Electricity dataset is presented in 5. Examples of unsupervised anomalies found in electricity dataset:

case: 17434

`vicdemand = 0.032626 [0.423±7.15E-8]`
`transfer = 0.500526 [0.41±6.96E-8]`
in rule:
`date` $> 0.0131 \wedge$ `nswprice` $\leq 0.0425 \rightarrow$ `UP`

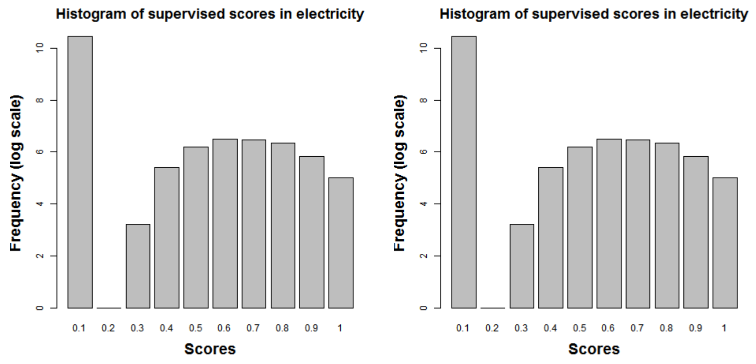

Fig. 5. Distribution of the anomaliness score (supervised and unsupervised) in the Electricity dataset.

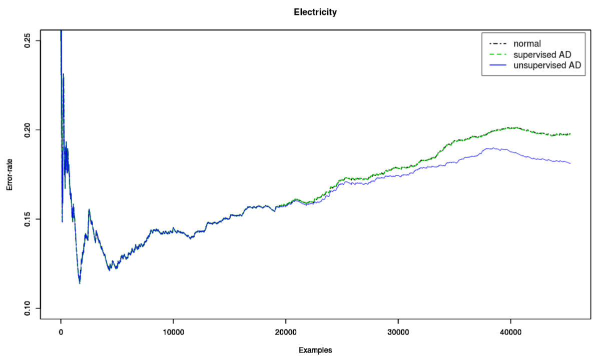

Fig. 6. Anomaly detection influence on prequential error in Electricity dataset

5 Conclusions

In this paper we present a one-pass, streaming algorithm for learning classification rules able to detect contextual anomalies in the data. Contextual anomalies are surprising attribute values in the context defined by the conditional part of the rule. The anomalies detected are characterized by a *context* that refers the region of the instance where the anomaly was detected, and behavioral attributes, those with anomalous values. For each example we compute the degree of anomaliness based on the probability of the attribute-values given the conditional part of the rule covering the example. Our system reports two types of anomalies: supervised and unsupervised anomalies. The examples with high degree of anomaliness are signaled to the user and not used to train the classifier. This is the main claim of this paper: online algorithms benefit from online anomaly detection by rejecting anomalous examples. The experimental evaluation in real-world data sets shows the ability to discover anomalous examples in well-known datasets. The main advantage of the proposed method is the ability to inform the context and explain why the anomaly occurs.

Acknowledgments

We thank the anonymous referees for their constructive comments. The authors acknowledge the financial support given by the project FCT-KDUS PTDC/ EIA/ 098355/2008 FCOMP -01-0124-FEDER-010053, the ERDF through the COMPETE Programme and by National Funds through FCT within the project FCOMP - 01-0124-FEDER-022701. Petr Kosina also acknowledges the support of Fac. of Informatics, MU, Brno.

References

1. José Barateiro and Helena Galhardas. A survey of data quality tools. *Datenbank-Spektrum*, 14:15–21, 2005.

2. Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), 2009.
3. Peter Clark and Robin Boswell. Rule induction with cn2: Some recent improvements. pages 151–163. Springer-Verlag, 1991.
4. Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In Raghu Ramakrishnan, Salvatore J. Stolfo, Roberto J. Bayardo, and Ismail Parsa, editors, *KDD*, pages 71–80. ACM, 2000.
5. A. Frank and A. Asuncion. UCI machine learning repository, 2010.
6. J. Gama, R. Fernandes, and R. Rocha. Decision trees for mining data streams. *Intelligent Data Analysis*, 10:23–45, 2006.
7. J. Gama, R. Rocha, and P. Medas. Accurate decision trees for mining high-speed data streams. In *Proceedings of the Ninth International Conference on Knowledge Discovery and Data Mining*. ACM Press, New York, NY, 2003.
8. João Gama and Petr Kosina. Learning decision rules from data streams. In Toby Walsh, editor, *IJCAI*, pages 1255–1260. IJCAI/AAAI, 2011.
9. Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2012.
10. M. Harries, C. Sammut, and K. Horn. Extracting hidden context. *Machine Learning*, 32:101–126, 1998.
11. V.J. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.
12. Elena Ikonomovska, João Gama, and Saso Dzeroski. Learning model trees from evolving data streams. *Data Min. Knowl. Discov.*, 23(1):128–168, 2011.
13. Edwin M. Knorr, Raymond T. Ng, and Vladimir Tucakov. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3-4):237–253, February 2000.
14. Petr Kosina and João Gama. Handling time changing data with adaptive very fast decision rules. In Peter A. Flach, Tijl De Bie, and Nello Cristianini, editors, *ECML/PKDD (1)*, volume 7523 of *Lecture Notes in Computer Science*, pages 827–842. Springer, 2012.
15. Duc-Son Pham, Svetha Venkatesh, Mihai Lazarescu, and Saha Budhaditya. Anomaly detection in large-scale data stream networks. *Data Mining and Knowledge Discovery*, to appear.
16. J. Ross Quinlan. Kdd-99 panel on last 10 and next 10 years. *SIGKDD Explorations*, 1(2):62, 2000.
17. W. Nick Street and YongSeog Kim. A streaming ensemble algorithm (sea) for large-scale classification. In *KDD*, pages 377–382, 2001.
18. John W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.
19. Indre Zliobaite and Bogdan Gabrys. Adaptive preprocessing for streaming data. *IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints):1, 2012.