Proceedings of Machine Learning Research 71:85–94, 2017 KDD Workshop on Anomaly Detection in Finance

# Fraud Detection with Density Estimation Trees

**Parikshit Ram** P.RAM@GATECH.EDU
*Atlanta, GA*

**Alexander G. Gray** AGRAY@CC.GATECH.EDU
*Palo Alto, CA*

## Abstract

We consider the problem of anomaly detection in finance. An application of interest is the detection of first-time fraud where new classes of fraud need to be detected using unsupervised learning to augment the existing supervised learning techniques that capture known classes of frauds. This domain usually has the following requirements – (i) the ability to handle data containing both numerical and categorical features, (ii) very low latency real-time detection, and (iii) interpretability.

We propose the use of a variant of density estimation trees (DETs) (Ram and Gray, 2011) for anomaly detection using distributional properties of the data. We formally present a procedure for handling data sets with both categorical and numerical features while Ram and Gray (2011) focused mainly on data sets with all numerical features. DETs have demonstrably fast prediction times, orders of magnitude faster than other density estimators like kernel density estimators. The estimation of the density and the anomalousness score for any new item can be done very efficiently. Beyond the flexibility and efficiency, DETs are also quite interpretable. For the task of anomaly detection, DETs can generate a set of decision rules that lead to high anomalous-ness scores. We empirically demonstrate these capabilities on a publicly available fraud data set.

**Keywords:** Anomaly detection, outlier detection, decision trees.

## 1. Financial fraud detection

The problem of fraud detection has a large impact in the field of finance. We consider the task of predicting whether a financial transaction or event in question is fraudulent or honest. In most cases, the fraudulent transactions make up a very small fraction of the total number of transactions. Given such a situation, there are two common ways of handling fraud detection. One way is to look at historical data where we have significant number of transactions which have been qualified as frauds and then treat this as a traditional binary classification problem (supervised learning). Another approach is to treat each fraudulent transaction as an outlier or anomaly because of it inherent rarity relative to the honest transactions and then treat the problem as an anomaly detection (AD) problem (unsupervised learning).

The favoured way of solving fraud detection is via binary classification (with some way to balance the classes in the classification loss function using techniques like stratified sampling or class-specific weights). An application of interest is the detection of first-time fraud where new kinds of fraud need to be detected using unsupervised learning to augment the existing supervised learning techniques that capture known kinds of frauds. Currently, in an usual real world setting, financial transactions constitute massive amounts of data, and there is a human monitoring team which screens transactions for possible first-time fraud via human-built rules. Any method aimed at facilitating or enhancing this setup will have the following requirements – (i) the ability to handle data containing both numerical and categorical features, (ii) very low latency real-time detection[1], and (iii) interpretability and auditability of the models and the predictions.

In this paper, we present preliminary results from our ongoing research for a solution which addresses the aforementioned requirements. Our solution is based on density estimation trees (DETs) (Ram and Gray, 2011). The rest of the paper is structured in the following way: In section 2 we present existing methods for anomaly and outlier detection and discuss why they are not ideal for fraud detection. Following that, we present our proposed solution in section 3 and evaluate the solution in section 4. We conclude this paper with some future directions and final remarks in section 5.

## 2. Outlier detection methods

Anomaly and outlier detection has been extensively studied in data mining literature with applications in many fields (Hodge and Austin, 2004; Chandola et al., 2009; Zimek et al., 2012). Among the unsupervised approaches, anomaly detection based on distributional properties of the data (such as density) is the most natural – examples with very low density or examples deemed to be beyond the support of the distribution are considered outliers. However, the estimation of densities and such quickly become hard and inaccurate in high dimensional data. Distance based approaches such as the one proposed in Ramaswamy et al. (2000) are widely used as a surrogate for density. Kernel machines based one-class support vector machine (OCSVM) (Schölkopf et al., 2001) performs outlier detection by estimating the high dimensional support of the data distribution and then classifying individual examples as being inside or outside that support. Neural networks based auto-encoders are another way of performing outlier detection. There are also various tree-based approaches such as isolation forest (Liu et al., 2008), RS-forest (Wu et al., 2014) and random projection forest (Chen et al., 2015).

These techniques usually only cater to numerical data. OCSVM would be able to handle both numerical and categorical data if a domain specific kernel function has been developed, but that is not a very common use case. Usually these methods require some categorical-to-numerical transformation such as one-hot encoding. The choice of the transformation is somewhat ad-hoc and they almost always remove the interpretability of the transformed feature. Moreover, most of these methods have a computationally expensive prediction operation – density-based methods and OCSVM require a kernel summation; distance-based approaches require a nearest-neighbour search; auto-encoders require a traversal of a potentially deep neural network. This makes it hard to get low latency real-time outlier detection. Decision trees based machine learning methods are usually known to handle mixed numerical and categorical features. However, the forest-based approaches to anomaly detection, such as isolation forest (Liu et al., 2008), would require non-trivial enhancements to the

[1] For example, the decision to flag a credit card transaction as an anomaly or not needs to be done in a very short period of time

algorithm[2] to handle categorical features in the native form. Furthermore, none of these methods (distance based, kernel machines based, neural networks based or forest based) are inherently interpretable. All these limitations make them a less-than-ideal solution for fraud detection.

## 3. Methodology

We take the approach of detecting outliers using distributional properties of the data (such as densities and probabilities) to identify anomalies. While we understand that not all outliers are anomalies, we are only focusing on the problem of outlier detection in this work. The main challenge here is to develop and use a meaningful distributional property that works with both numerical and categorical features. Probability mass function (PMF) can be used if all features are categorical (albeit with issues we will discuss in the ensuing subsection). Probability density functions (PDF) can be used to handle numeric features. We wish to develop an estimator that can handle both kinds of features. The approach we are considering is the conversion of numerical features to ordinal (ordered categorical) features and then defining a PMF over all the categorical variables (natively categorical as well as ordinal). However, estimating PMF with multiple categorical features (each with multiple categories) is a non-trivial task because of the large number of parameters involved.

We believe that the choice of converting numerical to ordinal (ordered categorical) has multiple advantages over the traditional categorical to numerical conversion. Firstly, usual categorical to numerical transformations, such as one-hot encoding, generate multiple numerical features for a single categorical feature. This makes it hard to judge (quantitatively) and interpret the effect of the original categorical feature in the final learned model. Secondly, decision tree based methods usually rely only on the ordering of the numerical features and not on the feature values themselves. By converting numerical values to ordinal values, we still approximately maintain this order by converting the original ordering to a partial ordering. There is some loss of information here for the resulting interpretability.

We present our decision tree based solution in the first subsection. In the following subsection, we formally describe how categorical features are naively handled, the issues that come with it and how we propose to handle those issues. We conclude this section with our proposal of converting numerical variables to categorical variables in the final subsection.

### 3.1. The decision tree

The decision tree (Breiman et al., 1984) is a widely used model for supervised learning. It has the following desiderata that fits our requirements for first time fraud detection:

- Since it is a combination of multiple, hierarchical univariate decisions, the decision tree model can seamlessly handle both categorical and numerical features,
- It allows for fast prediction times leading to low-latency real-time detection, and

[2] For example, in isolation forest, the splits required for the tree construction are generated by uniform randomly sampling within the range of the **numerical** feature, thereby separating outlier numerical feature values from the rest of the feature values. This form of split generation is hard to translate for categorical features.

- It provides an interpretable model in the form of (i) a tree visualization, and (ii) extractable rules to identify outliers.

The decision tree was used in an unsupervised setting in the form of DETs (Ram and Gray, 2011) for the task of estimating a probability density function (PDF). However, the primary focus in that paper was on numerical features since a PDF is only defined with numerical features. These density estimates could then be used to generate outlier scores for anomaly detection.

For data with categorical features, the quantity to estimate would a probability mass function or PMF. However, as we will describe in further detail in the next subsection, a PMF defined on a multi-dimensional categorical data is very hard to estimate and interpret. Instead, we propose representing a PMF in the form of a hierarchical decision tree with each internal node containing a univariate set membership based decision and each leaf node containing the estimate for the PMF for all point that filter to that leaf. Each node in this tree represents a subset of all possible categorical combinations in the data. The union of the sets of categorical combinations represented by all leaf nodes is equal to all possible categorical combinations. This makes the PMF more interpretable and allows us to leverage the inherent advantages of the decision tree.

For constructing such a tree, we will define the estimate for each leaf node and the splitting scheme with categorical features for each internal node in the next subsection. Given these two pieces, we will be able to build a decision tree in a top down manner upto a (user-specified) particular depth or number of leaf nodes or can be grown to purity, followed by a cost-complexity pruning (Breiman et al., 1984). Note that this is an **unsupervised** decision tree (that is, the tree is constructed without any labels or targets for the points), in contrast to the supervised formulation for classification or regression trees.

### 3.2. Categorical features

For a categorical variable with $k$ categories $C = \{c_1, \ldots, c_k\}$, the *categorical distribution* is defined as

$$p(x) = \prod_{i=1}^{k} p_i(x)^{\mathbb{I}(x=c_i)},$$

where $p_i = \Pr(x = c_i)$. This distribution has $k$ (possibly distinct) parameters (let us call this the *expanded PMF*). Alternately, the categories can be split into (say) 2 subsets $C_1$ and $C_2$ of $C$ such that $C_1 \cup C_2 = C$ and $C_1 \cap C_2 = \emptyset$ and set $p_i = p_{C_1}\ \forall c_i \in C_1$ and $p_i = p_{C_2}\ \forall c_i \in C_2$ for some $p_{C_1}, p_{C_2} \in [0, 1]$. In that case, the categorical distribution is now still defined by $k$ parameters but with *only two distinct values*, forming a *condensed PMF*. For estimation purposes with $n$ samples $S = \{X_1, \ldots, X_n\}$,

$$\hat{p}_i = \frac{\sum_{m=1}^{n} \mathbb{I}(X_m = c_i)}{n},$$

while

$$\hat{p}_{C_1} = \frac{\sum_{m=1}^{n} \mathbb{I}(X_m \in C_1)}{n|C_1|}.$$

We are essentially reducing the number of parameters by making different categories share parameters (and equally distributing the mass among categories sharing a parameter). We are making use of the condensed PMF for interpretability and estimation purposes. For an allowed number of parameters, the goal is to find the right grouping of the categories such that this condensed PMF is not that far off from the expanded PMF (in terms of say log-likelihood). One way to do this would a greedy top-down binary splitting of the categories into a hierarchical tree structure until we have the desired number of leaves (each leaf corresponding to a group of categories sharing a parameter).

This comes with a possible drawback – if a category (we are still discussing a single categorical feature) indicates that a point is an outlier (and hence possibly anomalous), this grouping of categories can potentially smudge that signal resulting in a case of *underfitting.*

Now consider a multi-dimensional categorical distribution with $d$ dimensions and $k_j$ categories $C_j = \{c_{j1}, \ldots, c_{jk_j}\}$ in each dimension $j = 1, \ldots, d$. The whole categorical distribution will then be defined by $\prod_{j=1}^{d} k_j$ parameters $\Pr(x_1 = c_{1i_1}, \ldots, x_d = c_{di_d})$ for each $i_j = 1, \ldots, k_j \; \forall j = 1, \ldots, d$.

Instead, if the categorical combinations are organized in a (possibly hierarchical) groups, this would provide some form of "regularization" and reduce the number of parameters. Let $N$ be one such group and let $C_j(N)$ be the categories of the $j^{\text{th}}$ categorical feature in the group $N$. Then the total categorical combinations that end up in $N$ is $\prod_{j=1}^{d} |C_j(N)|$ and all these categorical combinations share a single parameter (say)

$$p_N = \Pr(x_1 = c_1, \ldots, x_d = c_d) \forall (c_1, \ldots, c_d) \in C_1(N) \times \ldots \times C_d(N).$$

Hence the total number of parameters in this PMF is controlled by the (possibly user-specified) number of groups of categorical combinations. For the purposes of estimation, we would spread the probability mass in $N$ across all categorical combinations equally (potentially underfitting). With $n$ samples $S = \{X_1, \ldots, X_n\}$,

$$\hat{p}_N = \frac{\sum_{m=1}^{n} \mathbb{I}(X_m \in N)}{n \prod_{j=1}^{d} |C_j(N)|}.$$

Splitting the group $N$ in a decision-tree style top-down greedy fashion on a single feature into two groups $N_l$ and $N_r$, we would want to choose the feature which maximizes the (empirical) reduction of some loss function. If we consider the log-likelihood as our loss function of interest,

$$L(N) = \sum_{m=1}^{n} \mathbb{I}(X_m \in N) \log \hat{p}_N.$$

If $N$ is split along some feature $d'$ into $N_l(d')$ and $N_r(d')$ such $C_{d'}(N_l)$ and $C_{d'}(N_r)$ are the categories of the feature $d'$ in $N_l(d')$ and $N_r(d')$ respectively and $C_{d'}(N_l) \cup C_{d'}(N_r) = C_{d'}(N)$ and $C_{d'}(N_l) \cap C_{d'}(N_r) = \emptyset$. For estimation purposes, with $X_{md}$ denoting the $d^{\text{th}}$ feature of the point $X_m \in S$, let $n_{N_l}$, the number of points in $S$ that would end up in $N_l(d')$, be defined as

$$n_{N_l} = \sum_{m=1}^{n} \mathbb{I}(X_m \in N \wedge X_{md'} \in C_{d'}(N_l)),$$

with the number of points $n_{N_r}$ in $N_r(d')$ defined as

$$n_{N_r} = \sum_{m=1}^{n} \mathbb{I}(X_m \in N \wedge X_{md'} \in C_{d'}(N_r)) = \left( \sum_{m=1}^{n} \mathbb{I}(X_m \in N) \right) - n_{N_l}.$$

Then the log-likelihood of the resulting groups $N_l(d')$ and $N_r(d')$ are given by:

$$L(N_l(d')) = n_{N_l} \log \frac{n_{N_l}}{n|C_{d'}(N_l)| \prod_{j=1, j\neq d'}^{d} |C_j(N)|},$$

and

$$L(N_r(d')) = n_{N_r} \log \frac{n_{N_r}}{n|C_{d'}(N_r)| \prod_{j=1, j\neq d'}^{d} |C_j(N)|}.$$

We would want to solve the following problem to find the split of the group $N$ into two subgroups:

$$\max_{d'=1,\ldots,d} \max_{N_l(d'),N_r(d')} L(N_l(d')) + L(N_r(d')) - L(N). \tag{1}$$

Since the categories in a categorical feature usually do not have any inherent order, finding the best decision tree split for that feature can be a combinatorial problem – for a feature with $k$ categories, there are $k!$ possible ordering and all of them must be tried to find the best split. For regression, there exists an efficient heuristic that is proven to find the best split. For our purposes of identifying outliers, we propose the following intuitive splitting heuristic for a categorical feature:

- Sort the categories based on their frequency in the tree node which we are trying to split, with the most frequent category on the left and least frequent on the right.
- Given this ordering of the categories, we pick the split along this order which maximizes the gain in Equation (1).

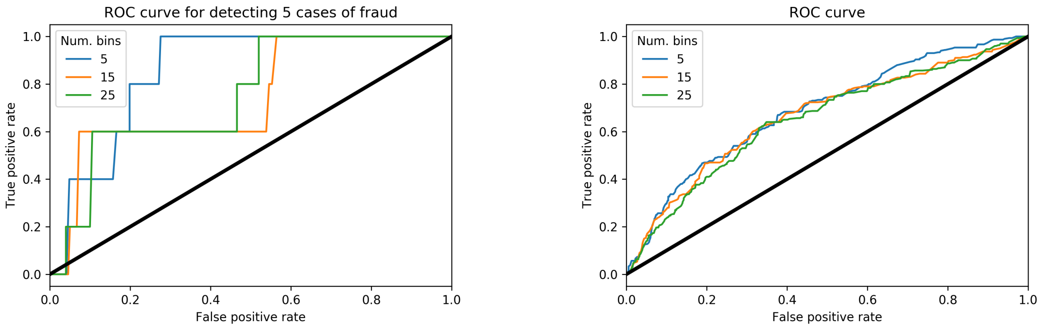
This heuristic is promoting the separation of points in regions of high probability mass in the left branch from points in regions of low probability mass in the right branch. This will attempt to separate outliers from the rest of the inliers. Moreover, at prediction time, we will handle unseen categories by filtering them down the right branch since the point contains a previously unseen category, implying that this point could potentially be an outlier relative to the training set.

### 3.3. Converting numerical features to ordinal features

A straightforward way of converting a numerical feature into an ordinal feature would be via quantization. We can then define a multi-dimensional categorical distribution (or a PMF) on a dataset which originally contained both categorical and numerical features and would now contain only contain categorical and ordinal variables post quantization. Here we consider standard one-dimensional quantization via uniform binning in the range of the numerical feature to a user-specified resolution. Note that, with these ordinal features, we pick the split along their inherent order while maximizing the gain in Equation (1).

## 4. Empirical evaluation

While we plan to comprehensively evaluate the proposed method on its estimation performance in our ongoing work, here we focus on the problem of outlier detection. To demonstrate the ability of the proposed scheme to handle both categorical and numerical



(*a*) 5 frauds, 700 non-frauds

(*b*) 300 frauds, 700 non-frauds

Figure 1: **ROC curves for detecting fraud cases.** For each of the 7 numerical features, we quantized the feature into 5, 15 and 25 ordinal values for each ROC curve.

features while maintaining their interpretabilty, we restrict ourselves to data sets containing *both* numerical and categorical features. While our proposed scheme will be seamlessly applicable to data sets with *all* categorical or *all* numerical columns, we believe that they are special cases of the situation we consider and have existing specialized schemes which we should consider as baselines. We plan to perform these evaluations in the ongoing research.

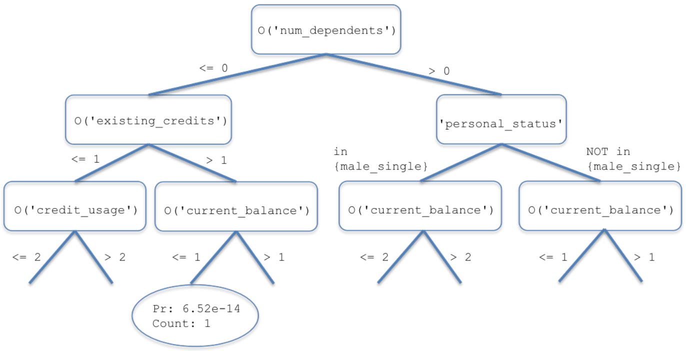


Figure 2: **Tree visualization of the top of a tree.** The feature on which a node is split is mentioned in the tree node. Numerical features that have been converted to an ordinal features are marked with an `O(·)`. For example, see the numerical features `num_dependents` and `credit_usage`. Categorical features such as `personal_status` are mentioned as is. The split decision is annotated on the corresponding branches and the leaf node contains the the number of points (`Count`) in it and then distribution function value (`Pr`) which is inversely proportional to the "outlier-ness" score.

We consider the German credit dataset from the UCI repository (Lichman, 2013) which contains 300 fraud and 700 non-fraud examples. The data contains 20 features – 13 categorical features and 7 numerical features[3]. Since 300 fraud cases in 1000 examples does not really represent a problem of outlier detection, we emulate that by using a dataset with 700 non-fraud examples and 5 randomly sampled fraud examples. These 705 examples are presented to our proposed method without any labels. On this dataset, we build an optimally pruned decision using the log-likelihood based notion of gain (Equation (1)) and utilizing the minimum cost complexity pruning with 10-fold cross-validation (Breiman et al., 1984).

Once the decision tree is constructed, the distribution function value for a point is used to determine its "outlier-ness". For any given set of points, the predicted distribution function values are sorted in the ascending order, with the lowest values predicted to be most likely to be outliers (and hence frauds). Using these scores, we generate ROC curves for any given set of points. Each evaluation is averaged over 10 runs. For numerical features, we utilize uniform binning with different number of bins as the quantization scheme. The ROC curves are presented in Figure 1. Figure 1($a$) corresponds to a test set with 5 fraud cases and 700 non-fraud cases, while figure 1($b$) corresponds to a test set with 300 fraud cases and 700 non-fraud cases.

These results are generated for different resolutions of quantization for each numerical feature. While the results indicate that our proposed scheme perform significantly better than random guessing, we are yet unable to figure out the optimal level of quantization for the purposes of outlier detection. Investigating the effect of the quantization scheme and resolution on the outlier detection performance is part of our ongoing research.

Beyond predictive performance, decision trees are also known for their interpretability. For example, Figure 2 presents a visualization of the learned decision tree for better interpretation of the model[4]. Moreover, we can extract rules corresponding to leaves in the decision tree that are most likely to contain outliers. For example, the following are rules (and some statistics) for 2 leaves in the tree most likely to contain outliers:

```
O(‘num_dependents’) > 0
  AND ‘personal_status’ NOT in set(["‘male single’"])
    AND O(‘current_balance’) > 1
- Count in leaf: 2
- Number of categorical combinations in leaf: 9.72e+11
- Distribution function value in leaf: 2.9e-15

O(‘num_dependents’) <= 0
  AND O(‘existing_credits’) > 1
    AND ‘credit_history’ NOT in set(["‘critical/other existing credit’"])
- Count in leaf: 1
- Number of categorical combinations in leaf: 2.6e+11
- Distribution function value in leaf: 5.43e-15
```

[3] We are restricted to a single data set mostly because we were unable to find publicly available fraud datasets which contain both categorical and numerical features.

[4] **Note.** We only present the top 3 levels of the tree. The trees tend to be quite deep and are hard to visualize within the confines of a manuscript. For the present tree, there was only a single leaf node that was reachable from the top 3 levels and hence is presented in Figure 2.

For each rule, we have added some statistics for the tree leaf corresponding to this rule. The numerical features are again marked with an `O(·)`. These rules allow the user to understand the decisions that went into flagging an example as a fraud. Moreover, these rules can be used as a SQL query in some database or datastream. These rules (and the tree visualization in Figure 2) demonstrate the ability of the learned model to utilize both numerical (`num_dependents`, `existing_credits`) and categorical (`personal_status`, `credit_history`) features.

## 5. Conclusion and future directions

In this paper, we presented our ongoing work on the development of an interpretable solution for fraud detection via outlier detection. We extended the decision-tree based DETs to handle numerical and categorical data at the same time. We achieve this by creating an estimator which can handle multiple categorical features and then proceed by pre-processing the numerical features into ordinal features and using the proposed estimator. While we believe that the initial results demonstrate the interpretability of the solution and show promise in terms of their predictive accuracy, we plan to evaluate the proposed solution more thoroughly across multiple axes against some existing baseline which can handle both numerical and categorical features at the same time. This includes the evaluation of the different quantization schemes such as uniform binning (which we presented here), quantiles, one dimensional k-means and Fayyad-Irani discretization (Fayyad and Irani, 1993).

More generally, given our strategy of converting numerical features to ordinal features, we wish to answer the following main questions:

- How to quantize the numerical feature?
- To what resolution should the numerical feature be quantized to?

To this end, we are also actively investigating estimators and training schemes which will allow us to perform the quantization of the numerical features as part of the training process to a level of resolution as required by the training. So instead of quantizing all numerical features in the same way to some (possibly arbitrary) level of resolution as a pre-processing step, we want to allow the model to have different quantization schemes and different resolutions for different numerical features as directed by the training process.

## References

Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

Fan Chen, Zicheng Liu, and Ming-ting Sun. Anomaly detection by using random projection forest. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 1210–1214. IEEE, 2015.

Usama Fayyad and Keki Irani. Multi-interval discretization of continuous-valued attributes for classification learning. 1993.

Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial intelligence review*, 22(2):85–126, 2004.

M. Lichman. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 413–422. IEEE, 2008.

Parikshit Ram and Alexander G Gray. Density estimation trees. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 627–635. ACM, 2011.

Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *ACM Sigmod Record*, volume 29, pages 427–438. ACM, 2000.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

Ke Wu, Kun Zhang, Wei Fan, Andrea Edwards, and S Yu Philip. Rs-forest: A rapid density estimator for streaming anomaly detection. In *Data Mining (ICDM), 2014 IEEE International Conference on*, pages 600–609. IEEE, 2014.

Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.