Classifier-Adjusted Density Estimation for Anomaly Detection and One-Class Classification

Lisa Friedland[*] Amanda Gentzel[*] David Jensen[*]

Abstract

Density estimation methods are often regarded as unsuitable for anomaly detection in high-dimensional data due to the difficulty of estimating multivariate probability distributions. Instead, the scores from popular distance- and local-density-based methods, such as local outlier factor (LOF), are used as surrogates for probability densities. We question this infeasibility assumption and explore a family of simple statistically-based density estimates constructed by combining a probabilistic classifier with a naive density estimate. Across a number of semi-supervised and unsupervised problems formed from real-world data sets, we show that these methods are competitive with LOF and that even simple density estimates that assume attribute independence can perform strongly. We show that these density estimation methods scale well to data with high dimensionality and that they are robust to the problem of irrelevant attributes that plagues methods based on local estimates.

1 Introduction

Anomaly detection, also known as outlier or novelty detection, is an active area of research with applications such as detecting fraud, errors, and unexpected categories in numerous domains [7,16]. Automated approaches to anomaly detection identify data instances with characteristics that appear inconsistent with the vast majority of the instances and thus are likely to have been generated by a different underlying process.

In the literature on anomaly detection, two challenges often arise. The first concerns merely agreeing on a task definition. This paper addresses the semi-supervised and unsupervised versions of the task [16], in which no labeled examples of anomalies are provided and the training data is assumed to be, respectively, either entirely normal ("positive" data), or normal mixed with a small number of unlabeled anomalies. The algorithms we discuss build a model of the training data, treating it as all positive, and then assign anomaly scores to points in a test data set. Our task can also be described as one-class classification [21], which is equivalent to semi-supervised anomaly detection without the restriction that the normal class be the most prevalent.

In the absence of labeled anomalies, a common task definition, which we adopt here, is to rank test points by their density according to a model of the positives, with the lowest density points being considered the most anomalous [5]. This definition puts the focus on globally modeling the positive data, setting aside concerns about whether the predicted anomalies are truly interesting to a domain specialist [25], how a point's density compares to that of its local neighborhood [6, 29], and what fraction of points to flag as anomalous [17,26].

The second challenge concerns the behavior of anomaly detection methods when faced with high-dimensional data, a topic of great interest recently [2, 14, 18, 19, 22, 33]. Most distance- and density-based outlier detection methods for multivariate data (e.g., DB-Outlier [17], local outlier factor [6]) rely on nearest-neighbor and distance computations, and they face difficulties when dimensionality increases. Zimek et al. [33] recently demonstrated that a key problem in high dimensions is irrelevant (or "noise") attributes, attributes that obscure the outlier properties of otherwise-anomalous points. Most approaches developed for high-dimensional data address this problem by working in lower-dimensional subspaces: they choose one or more low-dimensional projections of the data, identify outliers in the subspaces, and aggregate the results [2,14,19,22].

In this paper, we show that relatively simple and efficient methods can estimate the global probability density function, rather than approximating its properties locally. Directly estimating the probability density alleviates both of the above challenges. First, it enables a clear-cut task definition: the degree to which a point is an outlier is inversely proportional to its probability density. Second, this definition implies that when attributes are added to a data set, the global density function changes, so the ground truth ranking of points is reordered. As a special case of interest, when the new attributes are uniformly distributed, the global density is only multiplied by a constant, so—provided the density is modeled accurately—the ranking does not change.

The method this paper evaluates, which we refer to as classifier-adjusted density estimation (CADE), directly estimates the joint density of a data set. CADE

[*]School of Computer Science, University of Massachusetts Amherst. {lfriedl, agentzel, jensen}@cs.umass.edu.

constructs an initial density estimate, generates artificial anomalies from that distribution, and then uses those artificial anomalies to train a probabilistic classifier that corrects the initial distribution. Although CADE has been described previously [12, 13], its properties and performance have not been widely understood, and it has not been frequently used or compared to more recently developed alternatives. Ironically, in many cases this method has not even been recognized as performing density estimation. The main contributions of this paper are to:

- Identify the most effective combinations of CADE components. Specifically, we find that k-nearest neighbors and random forest classifiers work well across a number of initial density estimates. We confirm the surprising observation by Hempstalk et al. [13] that simple density estimates can perform strongly even when used alone, and we refute their conjecture that uniform initial density estimators would work poorly in high dimensions.

- Compare the performance of CADE and state-of-the-art anomaly detection algorithms in the presence of irrelevant attributes. Specifically, we show that as dimensionality increases, CADE is much more robust to irrelevant attributes than two local outlier factor (LOF) methods.

2 Classifier-Adjusted Density Estimation

CADE uses artificially generated data to transform a density estimation problem into a supervised learning problem. Hastie et al. [12] presented this idea more than a decade ago with the still-appropriate remark that although it "seems to have been part of the statistics folklore for some time, it does not appear to have had much impact despite its potential to bring well-developed supervised learning methodology to bear on unsupervised learning problems." We adopt the notation and derivation of Hempstalk et al. [13].

2.1 Derivation To use this technique, we must select two tools. First, we need an initial density estimation procedure for the positive data, T. This initial (or "naive") estimate need not be high quality; as experiments will show, a bounded uniform density or a single Gaussian can perform quite well. The only requirement is to be able to generate data from this distribution. We refer to the initial density as A, having probability density function $P(X|A)$. Second, we need a classifier—specifically, a conditional probability estimator that outputs class probability estimates (as opposed to binary predictions). The classifier will be trained to distinguish the positive data, T, from samples generated by the initial density estimate, A. We call these samples artificial anomalies or artificial negatives, cautioning that those terms can be misleading since the artificial points may closely resemble the positives.

Given a test example, the classifier outputs a probability, $P(C = T|X) = 1 - P(C = A|X)$, that the example comes from class T. CADE uses this probability estimate together with the initial density function $P(X|A)$ to compute a probability density for the positive class, $P(X|T)$.

To derive that computation, first we rewrite the classifier's estimate using Bayes' theorem, then we expand the denominator to express that X is from one of the two classes:

$$\begin{aligned} P(C = T|X) &= \frac{P(X|T)P(C = T)}{P(X)} \\ &= \frac{P(X|T)P(C = T)}{P(X|T)P(C = T) + P(X|A)P(C = A)} \end{aligned}$$

Next, we multiply out the terms and solve for $P(X|T)$:

$$\begin{aligned} P(X|T)P(C = T)P(C = T|X) \\ + P(X|A)P(C = A)P(C = T|X) \\ = P(X|T)P(C = T) \end{aligned}$$

(2.1)
$$P(X|T) = \frac{P(X|A)P(C = A)P(C = T|X)}{P(C = T)(1 - P(C = T|X))}$$

The left side of Equation (2.1) is the probability density estimate CADE produces for any desired point X. The right side consists of three components: the initial density estimate $P(X|A)$, a (constant) prior odds term $\frac{P(C=A)}{P(C=T)}$ describing the class ratio in the classifier's training data, and an odds term $\frac{P(C=T|X)}{1-P(C=T|X)}$ formed from the classifier's prediction. Throughout this paper, we train the classifier with equal-sized classes, so the prior odds term can be ignored.

Intuitively, the term from the classifier, which can range from 0 to ∞, can be seen as an adjustment factor to the initial density $P(X|A)$. The classifier has been trained to distinguish the true positives from the initial density (the artificial negatives). At one extreme, if the initial density estimate were perfect—if $P(X|A) = P(X|T)$ everywhere—then the classifier would not be able to distinguish A from T and would (ideally) predict $P(C = T|X) = 0.5$ everywhere. In that case, Eq. (2.1) simplifies to $P(X|T) = P(X|A)$, the initial density. At the other extreme, if the initial density estimate is uniform—if $P(X|A) = b$ for some constant b, over the range of all query points X—then Eq. (2.1) reduces to $P(X|T) \propto P(C = T|X)$. In that case, the final ranking is equivalent to the ranking from the classifier.

This paper builds on the work of Hempstalk et al. [13] and extends it in several ways. First, while Hempstalk et al. use Gaussians and mixtures of Gaussians for the initial density estimate, arguing that a uniform density would work poorly in high dimensions, we find good performance across a spectrum of initial densities from simple (uniform) to complex (kernel density estimate or Bayes net). Second, we explore a number of classifiers beyond the bagged decision trees (random forests) they use. Third, while Hempstalk et al. compare CADE to one-class SVMs [26], we show it is similarly competitive with LOF [6]. Finally, we show that CADE scales to high dimensions and large data sets, performing well in situations where other methods degrade.

2.2 Use of Artificial Anomalies Several previously described anomaly detection methods also use artificial anomalies as a negative class to train a classifier. These methods ignore the distribution of the artificial anomalies, either using the classifier's prediction alone [8, 10, 31] or wrapping it in a more complex procedure [1]. Using CADE (Eq. (2.1)) is probabilistically well-founded, but it is worth understanding what happens when using the classifier alone. Two questions arise: (1) how should the artificial anomalies be distributed, and (2) when will the test data be similar enough to the training data for the classifier to generalize correctly?

Assertions for where to put the artificial anomalies have ranged from covering the non-positive space as thoroughly as possible [10], to near the positive data but concentrated in its sparse regions [8], to matching the positives as closely as possible [13]. As an illustration, suppose we use a uniform distribution of artificial anomalies, defined as constant over some range. We train a classifier and then encounter a test point far outside this range. The classifier will make a prediction if asked, but since the point is unlike the training data, that prediction is unreliable. (Precisely such a concern in a supervised anomaly detection problem was the motivation for work by Bishop [5].) It would be preferable to directly recognize such a point and mark it as anomalous on the basis of being far from the training data. We could manually add handling for such cases, but Eq. (2.1) automatically provides this property: the classifier's prediction is multiplied by a constant if the new point is within the range of the artificial anomalies and by zero[1] outside that range, so points far from the positives are assigned the lowest density.

The same situation occurs with non-uniform initial densities: test points may be located outside the regions

[1]Multiplying probabilities by zero causes information to be lost, so we actually use a small positive value.

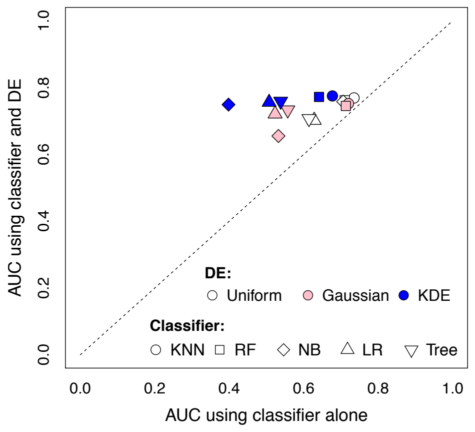

Figure 1: AUC improvement from adding density estimates (DE) to classifier predictions.

where the classifier was trained. By multiplying the classifier's prediction by $P(X|A)$, CADE produces low density estimates in places where artificial anomalies, and presumably training data itself, are absent.

It is striking how much improvement Eq. (2.1) provides over ranking by the classifier's output alone. Fig. 1 shows the average performance with and without density estimates, in the semi-supervised setting (see Sections 3 and 4). The AUCs are calculated in the same way as the overall averages of Table 2. On average, the density estimate improves upon the classifier alone for all 15 initial density estimate and classifier combinations.

3 Data, Tasks and Evaluation

We apply CADE to an assortment of real (Table 1) and synthetic data sets. We transform 15 data sets from the UCI repository [3] into anomaly detection problems by selecting certain class labels to be the positive data and treating the others as the negative, or anomalous, data. A final data set, *Employee*, bears special mention. It was collected as part of the DARPA ADAMS program and intended as a test bed for anomaly detection—namely, detection of insider threats to an organization's information systems [27]. It describes the online activities of ∼5500 employees at a large business. The data were collected using a commercial tool that monitors daily computer usage, recording events such as logons, websites visited, files printed and external devices connected. Every month, a small number of synthetically constructed "malicious user activities" were added to the data. There are 6 month-long data sets, each containing 88 numeric attributes and over 100,000 instances. The instances summarize (through normalized counts and ratios) the events recorded per user per day, and the class label indicates user-days in which malicious activities took place.

Table 1:
Characteristics of semi-supervised / one-class classification data sets

Name	Total instances	Number of attributes		Classes in raw data	Task instances in experiments		Smaller[b] class size, as % of total	Avg. runtime per fold (sec.)
		Numeric	Nominal		S[a]	M[a]		
Adult	48,841	6	8	2	2	–	24%	56.1
Ann-thyroid	3,772	15	6	3	3	3	2–92%	17.6
Bands	540	21	14	2	2	–	35%	5.6
Breast cancer	699	9	–	2	2	–	35%	8.0
Contraceptive	1,473	2	7	3	3	3	23–43%	11.1
Credit	690	6	9	2	2	–	46%	6.9
Ecoli	336	7	–	8	2	2	23–42%	5.7
Glass	214	9	–	6	3	3	13–36%	6.9
Ionosphere	351	34	–	2	2	–	36%	7.2
Musk	6,598	166	–	2	2	–	14%	28.1
Pendigits	10,992	16	–	10	10	10	10%	20.6
Segment	2,310	19	–	7	7	7	14%	11.5
Yeast	1,484	8	–	10	4	4	11–31%	7.9

Characteristics of unsupervised anomaly detection data sets

Name	Instances	Numeric attrs.	Nominal attrs.	Classes in raw data	S	M	Smaller class size	Avg. run-time (sec.)
Coil	4,000	82	2	2	2	–	1.3%	21.0
Shuttle[c]	58,000	9	–	7	5	–	0.02–7%	104.3
Employee (6 versions)	108,215–133,770	88	–	2	6	–	0.006–0.08%	368.1

[a] S = number of binary class divisions formed using a single class as the positives; M = number of binary class divisions formed using the union of multiple classes as the positives. [b] Or range of raw class sizes, when S > 2.
[c] With *Shuttle*, we reproduce the setup of Lazarevic and Kumar [19], in which class 1 is always used as positive, and each of 2, 3, 5, 6, and 7 in turn are used as negatives.

The largest set of experiments takes place in a semi-supervised, or one-class classification, setting. From 13 of the UCI data sets, we create a total of 76 class divisions into positive and anomalous data: first, each class label is used as a single positive class, with the union of the other classes labeled as anomalies; next, each class label is used as a single anomalous class, with the union of the other classes labeled as positives[2]. In this one-class setting, we run 10-fold cross validation: in each of 10 runs, 90% of the data is used for training and 10% for testing. At training time, only positive instances from the training set are seen, and an equal number of artificial anomalies are created. At evaluation time, all of the test instances, both positive and negative, are scored. Thus the class proportions in the test data match the proportions in the full data set. These vary widely, as shown in Table 1.

In many real-world settings, such as with *Employee*, it is unrealistic to assume that a clean set of positives is available at training time. We address this scenario with a separate set of experiments in an unsupervised setting. In this setting, we sample 10,000 unlabeled instances as training data[3], and at test time, we score all instances. This setup is used for *Employee* and the remaining 2 UCI data sets.

Although the outputs of CADE are probability density estimates, we do not examine the individual scores, only their ranking. This ranking is compared to the true class labels using area under the ROC curve (AUC), which varies between 0.5 (random) and 1 (perfect). This measure is chosen because it reflects how well the ranking separates the positives from the negatives and because it does not vary as a function of class proportions [9].

4 Choosing Components of CADE

4.1 Methods Implemented We use four types of initial density estimates for numeric data: uniform, Gaussian, kernel density estimate, and Bayes net. The

[2]We discard class divisions having fewer than 50 instances in either the positives or the negatives because, when used with cross validation, the test sets are often too small to be meaningful.

[3]For runtime reasons specific to this implementation, we use only 2,000 with the Bayes net.

uniform density is defined as a constant over the range of the training data. The Gaussian is a single Gaussian with diagonal covariance, or equivalently, a product of independent Gaussians, one distribution fit to each dimension. The kernel density estimator (KDE, also known as a Parzen-Rosenblatt window) [24] is a flexible, nonparametric alternative we use in a similar way: fit a univariate KDE to each dimension independently, and define the joint density to be the product of the independent marginal densities. The kernel itself is Gaussian, and the bandwidth in each dimension is chosen using a plug-in selector. Since the first three density estimates cannot capture dependencies among attributes, we compare them to a full joint model, a Bayesian network learned over the training data. We expect this alternative to be more powerful but computationally expensive. Further implementation details and recommendations are available in Sections 8–9[4].

As classifiers within CADE, we experiment with five methods available in Weka [11]: k-nearest neighbors (KNN), random forest (RF), naive Bayes (NB), logistic regression (LR), and decision trees (Tree—alternatively CART or C4.5). We also use the density estimates alone, unadjusted by classifiers. Most results are reported only on the best-performing classifiers, KNN and RF. Among the others, Tree always performs worse than random forest, and naive Bayes and logistic regression are ineffective if used with KDEs, as Section 9.1 explains.

4.2 Performance on Real Data The top portion of Table 2 summarizes the results of one-class classification. Each method is run with 10-fold cross validation across a total of 76 class divisions formed from the 13 data sets. Each entry represents the average AUC across the class divisions of a single data set. For data sets with more than two class labels, we subdivide the class divisions into single-class (S) versus multi-class (M) positives, since the multi-class positives are more complex distributions to learn and generally yield lower AUC. The overall average for each method is calculated by averaging all rows. The column labeled "Supervised RF" shows the performance of a random forest that is trained using fully labeled positives and negatives. This value should be an indicator of problem difficulty and an approximate ceiling to the performance of CADE.

We find considerable variation among data sets and class divisions regarding the relative performance of the CADE versions. Some versions, such as Uniform + RF and KDE, are consistently among the best. It is interesting to note that using more complex density estimates (KDE, Bayes net) does not consistently improve on using a uniform density estimate. One surprising result is how well the unadjusted density estimates can perform. KDE is frequently one of the highest performers, and even an unadjusted uniform density estimate (which essentially draws a box around the positive data and labels anything outside the box as anomalous) can give high AUC in many cases. While no method is a top performer for all data sets, Uniform + RF and KDE + RF seem to be practical, simple choices when selecting which version of CADE to use.

In the unsupervised setup (Table 2 bottom), there is again considerable variation among CADE versions. The initial density estimates again perform well, often just as well as their classifier-adjusted counterparts. In these experiments, the CADE versions that use Bayes net are frequently among the strongest performers. Bayes net's ability to learn a full joint model may give it this edge, although of course it is also the most computationally expensive.

4.3 Effect of Correlation It is counterintuitive that initial density estimates that do not model attribute dependence would be as effective as we have seen. We expect that adjusting the density estimates to capture dependence should be beneficial, particularly when the distributions do have high correlation. To test this hypothesis, we compare the performance of full CADE to that of an unadjusted density while varying the correlation of synthetic data.

When using synthetic data, we can compute the ground truth ranking of any set of test instances with respect to the probability density function of the positive data. In this experiment, in each trial, we randomly generate a different five-dimensional Gaussian, generate positive data from it, and run CADE to estimate its distribution. Using a test set of points drawn from the same Gaussian, we compute the points' CADE-estimated densities and their true densities. We compare the two rankings using Spearman's rank correlation coefficient (also known as Spearman's ρ), a nonparametric measure of correlation that ranges from -1 to 1. We have not previously seen Spearman's rank correlation used to evaluate anomaly detection, but we recommend it as an objective measure that can be used with any test set whenever the true distribution is known.

Fig. 2 shows the results of 400 such trials. To measure the dependence in the data, we compute the average absolute value of the off-diagonal entries from the Gaussian's correlation matrix, and plot this along the x axis. We can see that, as the correlation increases, the performance gap between adjusted and unadjusted KDE widens. A similar effect is observed for other

[4]Sections 8–10 are provided as Supplemental Materials at `http://kdl.cs.umass.edu/papers/friedland-et-al-sdm2014-supplemental.pdf`.

Table 2:
Semi-supervised / One-class classification results (Each entry: AUC averaged across cross-validation folds and class divisions)

Name	Supervised	Uniform			Gaussian			KDE			Bayes Net			LOF	
	RF	none	KNN	RF	none	KNN	RF	none	KNN	RF	none	KNN	RF	LOF	Bagged
Adult (S)	0.673	*0.515*[b]	0.637	0.673	**0.700**[a]	**0.695**	**0.699**	**0.698**	***0.690***	*0.689*	0.658	0.663	0.657	*0.520*	*0.544*
Ann-thyroid S	0.998	0.851	0.877	**0.979**	*0.812*	*0.795*	*0.844*	0.951	0.950	0.953	*0.763*	*0.762*	*0.862*	*0.765*	*0.767*
Ann-thyroid M	0.998	*0.729*	*0.778*	**0.943**	*0.693*	*0.681*	*0.793*	0.842	*0.841*	0.844	*0.675*	*0.680*	*0.859*	*0.671*	*0.681*
Bands (S)	0.768	0.559	**0.618**	**0.623**	0.599	0.601	0.609	**0.623**	**0.623**	**0.626**	0.553	0.553	0.557	0.570	0.556
Breast cancer (S)	0.993	*0.689*	0.940	0.895	**0.982**	**0.982**	**0.978**	**0.980**	**0.980**	**0.977**	0.918	0.918	0.908	0.943	0.820
Contraceptive S	0.701	*0.514*	0.554	0.561	0.558	0.553	0.556	**0.583**	**0.577**	**0.578**	0.554	0.558	**0.581**	0.508	*0.501*
Contraceptive M	0.702	0.500	0.509	**0.522**	0.511	0.505	0.512	**0.529**	**0.521**	**0.524**	0.506	0.510	**0.526**	0.513	0.505
Credit (S)	0.932	*0.520*	**0.819**	0.715	0.792	0.797	0.784	0.692	0.695	0.693	0.773	0.774	0.757	0.796	0.608
Ecoli S	0.975	0.849	0.915	0.929	**0.943**	**0.942**	**0.945**	0.935	0.935	0.935	0.935	0.934	0.935	0.932	**0.939**
Ecoli M	0.976	*0.583*	0.819	0.658	0.782	0.800	0.789	0.846	0.852	**0.867**	0.835	0.845	0.838	0.841	0.846
Glass S	0.930	0.699	0.744	0.734	0.734	0.740	0.744	0.751	0.753	0.756	**0.767**	**0.768**	**0.767**	**0.777**	**0.770**
Glass M	0.923	0.519	*0.585*	0.620	0.597	0.603	0.593	0.616	0.617	0.624	0.605	0.613	0.610	**0.691**	0.661
Ionosphere	0.968	0.724	0.610	**0.839**	0.615	0.616	0.617	**0.835**	**0.835**	**0.836**	0.614	0.616	0.615	0.597	0.610
Musk	0.992	*0.705*	*0.837*	0.796	*0.593*	*0.592*	*0.604*	0.810	0.809	0.813	0.657	0.664	0.665	**0.882**	**0.873**
Pendigits S	1.000	*0.813*	*0.979*	*0.985*	*0.953*	*0.973*	*0.969*	*0.962*	*0.973*	*0.974*	**0.988**	**0.988**	**0.988**	**0.996**	**0.996**
Pendigits M	1.000	*0.500*	*0.907*	*0.848*	*0.613*	*0.858*	*0.702*	*0.668*	*0.875*	*0.805*	*0.742*	*0.848*	*0.790*	**0.983**	**0.976**
Segment S	0.999	0.948	0.959	**0.970**	0.944	0.948	0.949	0.959	0.961	0.961	**0.972**	**0.973**	**0.973**	**0.974**	**0.973**
Segment M	0.999	*0.648*	0.891	0.775	*0.594*	*0.747*	*0.633*	*0.758*	*0.834*	*0.799*	*0.712*	*0.759*	*0.746*	0.909	**0.922**
Yeast S	0.865	*0.638*	0.725	0.720	0.754	**0.765**	0.752	0.725	0.729	0.729	0.754	**0.755**	0.750	0.741	0.744
Yeast M	0.866	0.505	0.582	0.557	*0.554*	*0.560*	*0.552*	0.530	0.550	0.556	*0.574*	*0.572*	*0.573*	**0.629**	**0.620**
Average S	0.907	0.694	0.786	0.801	0.768	0.769	0.773	0.808	0.808	**0.809**	0.762	0.763	0.770	0.769	0.746
Average M	0.923	0.569	0.724	0.703	0.621	0.679	0.653	0.684	0.727	0.717	0.664	0.690	0.706	**0.748**	0.745
Overall Average	0.913	0.650	0.764	0.767	0.716	0.738	0.731	0.765	**0.780**	**0.777**	0.728	0.738	0.748	0.762	0.746

Unsupervised results (Each entry: AUC averaged across 10 runs)

Coil		*0.500*[c]	*0.843*	**0.940**	*0.909*	*0.906*	*0.913*	*0.811*	*0.830*	*0.809*	*0.910*	*0.910*	*0.913*	*0.805*	*0.825*
Shuttle (1 vs. 2)		*0.500*	*0.984*	**0.998**	*0.853*	*0.963*	*0.962*	*0.978*	***0.990***	***0.991***	0.977	0.982	**0.994**	0.986	**0.993**
Shuttle (1 vs. 3)		*0.500*	*0.919*	**0.996**	*0.936*	*0.868*	*0.977*	*0.967*	*0.884*	*0.976*	0.967	0.960	**0.991**	0.957	0.960
Shuttle (1 vs. 5)		*0.500*	*0.848*	0.930	**0.998**	*0.956*	***0.998***	*0.986*	*0.556*	*0.980*	0.985	0.984	0.985	*0.497*	*0.450*
Shuttle (1 vs. 6)		*0.500*	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	***0.997***	***0.998***	***0.998***	**1.000**	**1.000**	**1.000**	**0.998**	**0.998**
Shuttle (1 vs. 7)		*0.500*	**1.000**	**1.000**	**0.999**	**0.999**	**0.999**	***0.990***	0.985	***0.993***	**0.999**	**0.999**	**0.999**	**0.999**	**0.999**
Employee Sep		*0.500*	*0.871*	**0.959**	*0.927*	*0.926*	*0.930*	***0.960***	**0.961**	**0.961**	*0.923*	*0.924*	*0.927*	*0.592*	*0.588*
Employee Oct		*0.500*	*0.870*	**0.978**	*0.938*	*0.938*	*0.941*	***0.973***	***0.973***	***0.973***	*0.940*	*0.940*	*0.943*	*0.651*	*0.647*
Employee Nov		*0.500*	**0.753**	*0.675*	*0.708*	*0.709*	*0.708*	*0.727*	*0.729*	*0.730*	**0.761**	***0.759***	**0.761**	*0.648*	*0.625*
Employee Dec		*0.500*	0.790	**0.820**	*0.778*	*0.778*	*0.780*	***0.820***	***0.819***	**0.821**	0.806	0.808	0.809	*0.561*	*0.549*
Employee Jan		*0.500*	*0.764*	*0.701*	*0.682*	*0.689*	*0.688*	*0.664*	*0.667*	*0.667*	***0.803***	**0.805**	***0.800***	*0.554*	*0.541*
Employee Feb		*0.500*	0.721	*0.483*	*0.582*	*0.583*	*0.580*	*0.556*	*0.561*	*0.561*	**0.733**	**0.735**	***0.729***	*0.537*	*0.526*
Average		0.500	0.860	0.873	0.859	0.860	0.873	0.869	0.829	0.872	0.900	0.900	**0.904**	0.732	0.725

[a] For each data set, the highest AUC is reported in bold, as are all scores within 0.01 of it. [b] Italics indicate methods that perform significantly worse than the best for each data set. Statistical significance is determined by performing paired t-tests between the 10 runs of a single class division using $\alpha = 0.001$. For the cross-validated results, a method is italicized if it is significantly worse than the highest performer in a majority of that data set's class divisions and is never significantly better. [c] The consistent 0.5 AUC for Uniform is a result of the unsupervised setup: here, the range of the uniform distribution is determined from the entire data set, so no test instance is ever outside that range.

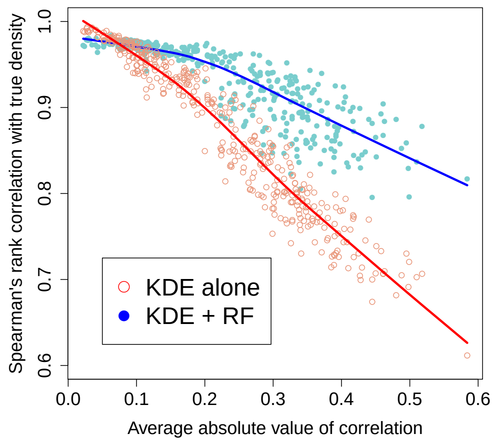

Figure 2: Adjusted and unadjusted KDE performance as a function of correlation. The lines are the locally weighted regression lines of the points.

naive density estimates and classifiers. This gap affirms the contribution of the classifier when attributes are correlated.

5 Scalability to High Dimensions and Large Data

One of the primary benefits of estimating a global probability density is the sensible behavior as dimensionality increases. To show that CADE's density estimation is a viable solution in high dimensions, we compare it to two popular outlier detection methods, showing that they perform comparably in UCI but that CADE has an advantage in synthetic and in some of the highest-dimensional data. As comparison methods, we use LOF [6] (available in Weka) and bagged LOF [19]. LOF is a popular anomaly detection method for multi-dimensional data, and it has spurred many variants. It computes a local density estimate by essentially comparing the size of a point's neighborhood to those of its neighbors [32].

As Zimek et al. discuss, most distance- and density-based methods are challenged if there are dimensions in which a point does not look like an outlier [33]. These are described as "irrelevant" attributes, of which the simplest possible are those with uniform distributions. Bagged LOF is an ensemble-style method that addresses this challenge by detecting outliers that only become apparent in subspaces of their native high-dimensional spaces. It runs multiple iterations of LOF, each with a randomly chosen subset of attributes, and then, for each instance, aggregates its scores from across the runs. Bagged LOF has been shown to be more resistant than LOF to irrelevant attributes, but only when the number of irrelevant attributes is low [19].

CADE is competitive with the LOF methods in the semi-supervised experiments of Table 2. Each of LOF and bagged LOF wins on more data sets than any single version of CADE, but there are also data sets where the LOF methods are far below the best performer. In the data sets run in unsupervised mode, CADE is stronger: several versions of CADE (Uniform + RF, KDE + RF, Bayes net, Bayes net + RF) outperform LOF on all but one or two data sets. On *Coil* and *Employee*, both with over 80 attributes, LOF is substantially below the maximum.

5.1 Robustness to Noise Attributes To test robustness to irrelevant attributes, we generate synthetic data with varying numbers of uniform attributes added on. In this experiment, the positive data is a mixture of three five-dimensional Gaussians with means and covariance matrices randomly sampled at each trial. In addition to the five informative attributes, between 0 and 80 uniformly sampled attributes are added. At test time, the instances are drawn from the same distribution as the positives[5], and Spearman's ρ is calculated between the true and the estimated rankings of densities. Figs. 3 and 4 show the (smoothed) average performance and 95% confidence intervals, across 20 trials per setting, for most versions of CADE and LOF.

While LOF and bagged LOF perform well initially, as we add uniform noise attributes, their performance drops off dramatically. Uniform + KNN has a rapid decline similar to LOF's. Some versions of CADE, however, do not experience the same drop-off. The top performers, Uniform + RF and KDE + RF, decline only gradually with noise attributes. As Fig. 4 shows, as noise attributes are added, the effect of the classifier drops off and the performance converges to that of the plain density estimate. Prior to that convergence, random forest outperforms k-nearest neighbors, and both outperform the density estimate alone.

5.2 Scalability CADE scales well to large numbers of instances and attributes. Table 1 displays the average number of seconds for a single run using KDE + RF[6]. On the *Employee* data sets, with 88 attributes and over 100,000 instances, CADE runs in under 7 minutes. The computational complexity depends on the implementations of the components used. Our KDEs use binning, which makes them in $O(gmn_{train})$ to learn and $O(gmn_{test})$ to apply, where m is the number of attributes, g is the grid size and n_{train} is the number of training instances. Random forests with balanced trees

[5]See Section 10 for an alternative test set.

[6]Due to details of our implementation, these numbers exclude the time to learn the KDE's bandwidth (< 10 sec.) and to load the data file.

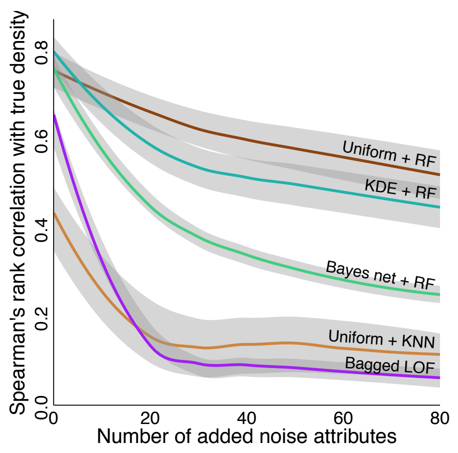

Figure 3: Performance as uniform noise attributes are added to a distribution with five informative attributes. LOF is similar to bagged LOF and is omitted for clarity.

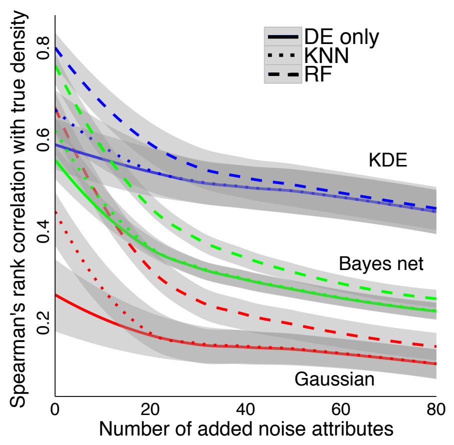

Figure 4: Same experiment as Fig. 3. Each group of three lines represents a single density estimation method. Line styles denote classifiers.

can be learned in $O(t \log(m) n_{train} \log(n_{train}))$ time, where t is the number of trees and $\log(m)$ attributes are considered at each split, and applied in $O(\log(n_{test}))$.

6 Related Work

Recent research in anomaly detection has often overlooked approaches for joint probability density estimation. Although statistical outlier detection is well-developed in the univariate case [4] and surveys have described many multivariate techniques [7, 16, 20], there seems to be a widespread assumption that estimating useful models of multivariate densities is infeasible. Common wisdom holds that parametric models such as Gaussians are unlikely to fit the data and that nonparametric models—namely KDEs [24]—become impractical in high dimensions. According to this reasoning, with KDEs, either the training data become too sparse, or the computational complexity (in naive implementations) gets too high [16, 20, 30]. Our KDE model sidesteps these issues by independently combining a set of one-dimensional KDEs (and combining the resulting initial density estimate with a probabilistic classifier).

The algorithms LOF and bagged LOF are representatives of what are called distance- and density-based outlier detection methods. Other variations include DB-Outlier [17], LOCI (local correlation integral) [23], and COF (connectivity-based outlier factor) [28]. In high dimensions, these approaches usually involve projecting the data into low-dimensional subspaces and performing traditional anomaly detection there. One of the earliest such methods searches for subspaces which contain unusually sparse cubes [2]. While bagged LOF creates random subspaces in which anomalies might be detectable [19], other recent methods focus on choosing useful subspaces [18, 22] and combining the subspace anomaly scores—potentially from spaces of different dimensionalities—in principled ways [14, 22].

Other techniques commonly used for anomaly detection come out of one-class classification. Two popular methods are support vector data description [30] and one-class SVM [26]. A final related technique is that of density ratio estimation [15]. Like CADE, this method compares two densities, but instead of training data and artificial anomalies, they are training data and test data. Instead of comparing them by learning a classifier, it applies techniques specialized for estimating ratios of densities.

7 Conclusion

This paper shows how to use classifier-adjusted density estimation to achieve good results at anomaly detection tasks. Contrary to conventional expectations, we show that density estimation can be a reasonable goal for high-dimensional data, giving results competitive with LOF over a large number of data sets and, unlike LOF, performing robustly even when faced with 80 irrelevant attributes. We find that CADE can work well with a variety of initial density estimate components and classifiers and that, depending on the data set, marginally independent density estimates can perform strongly even before adjustment by a classifier.

Acknowlegments

The authors thank Tom Dietterich for initially suggesting that CADE-style methods deserved greater study. Funding was provided by the U.S. Army Research Office and Defense Advanced Research Projects Agency 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.

References

[1] N. Abe, B. Zadrozny, and J. Langford. Outlier detection by active learning. In *ACM SIGKDD 2006*, pp. 504–509.

[2] C. C. Aggarwal and P. S. Yu. Outlier detection for high dimensional data. In *ACM SIGMOD 2001*, pp. 37–46, New York, NY.

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

[4] V. Barnett and T. Lewis. *Outliers in Statistical Data.* Wiley, Chichester; New York, 1994.

[5] C. M. Bishop. Novelty detection and neural network validation. *IEE Proceedings - Vision, Image and Signal Processing*, 141(4):217–222, 1994.

[6] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *ACM SIGMOD 2000*, pp. 93–104, New York, NY.

[7] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):1–58, July 2009.

[8] 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*, pp. 123–130. IEEE.

[9] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[10] F. A. González and D. Dasgupta. Anomaly detection using real-valued negative selection. *Genetic Programming and Evolvable Machines*, 4(4):383–403, 2003.

[11] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, Nov. 2009.

[12] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* New York: Springer-Verlag, 2001.

[13] K. Hempstalk, E. Frank, and I. Witten. One-class classification by combining density and class probability estimation. In *ECML/PKDD 2008*, volume I, pp. 505–519. Springer.

[14] M. Henrion, D. J. Hand, A. Gandy, and D. J. Mortlock. CASOS: A subspace method for anomaly detection in high dimensional astronomical databases. *Statistical Analysis and Data Mining*, 6(1):53–72, 2013.

[15] S. Hido, Y. Tsuboi, H. Kashima, M. Sugiyama, and T. Kanamori. Statistical outlier detection using direct density ratio estimation. *Knowledge and Information Systems*, 26(2):309–336, 2011.

[16] V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[17] E. M. Knorr, R. T. Ng, and V. Tucakov. Distance-based outliers: Algorithms and applications. *The VLDB Journal*, 8(3–4):237–253, 2000.

[18] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek. Outlier detection in axis-parallel subspaces of high dimensional data. In *PAKDD 2009*, pp. 831–838. Springer-Verlag.

[19] A. Lazarevic and V. Kumar. Feature bagging for outlier detection. In *ACM SIGKDD 2005*, pp. 157–166, New York, NY.

[20] M. Markou and S. Singh. Novelty detection: A review—Part 1: Statistical approaches. *Signal Processing*, 83(12):2481–2497, 2003.

[21] M. Moya, M. Koch, and L. Hostetler. One-class classifier networks for target recognition applications. Technical report, Sandia National Labs, Albuquerque, NM, 1993.

[22] E. Müller, M. Schiffer, and T. Seidl. Statistical selection of relevant subspace projections for outlier ranking. In *ICDE 2011*, pp. 434–445, Washington, DC, USA. IEEE Computer Society.

[23] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos. LOCI: Fast outlier detection using the local correlation integral. In *ICDE 2003*, pp. 315–326. IEEE, 2003.

[24] E. Parzen. On the estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[25] D. Pelleg and A. Moore. Active learning for anomaly and rare-category detection. *Advances in Neural Information Processing Systems*, 18(16):1073–1080, 2004.

[26] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 13(7):1443–1471, July 2001.

[27] T. E. Senator, H. G. Goldberg, A. Memory, and others. Detecting insider threats in a real corporate database of computer usage activity. In *ACM SIGKDD 2013*, pp. 1393–1401.

[28] J. Tang, Z. Chen, A. W.-C. Fu, and D. W.-L. Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *PAKDD 2002*, pp. 535–548, London, UK. Springer-Verlag.

[29] L. Tarassenko, P. Hayton, N. Cerneaz, and M. Brady. Novelty detection for the identification of masses in mammograms. In *Fourth International Conference on Artificial Neural Networks*, pp. 442–447, 1995.

[30] D. M. Tax and R. P. Duin. Support vector data description. *Machine learning*, 54(1):45–66, 2004.

[31] J. P. Theiler and D. M. Cai. Resampling approach for anomaly detection in multispectral images. In *SPIE 2003*, volume 5093, pp. 230–240.

[32] V. Vintrova, T. Vintr, and H. Rezankova. Comparison of different calculations of the density-based local outlier factor. In *IMMM 2012*, pp. 60–67.

[33] 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, Oct. 2012.