

Online Density Estimation of Heterogeneous Data Streams in Higher Dimensions

Michael Geilke^(✉), Andreas Karwath, and Stefan Kramer

Johannes Gutenberg University Mainz, Mainz, Germany
{geilke,karwath,kramer}@informatik.uni-mainz.de

Abstract. The joint density of a data stream is suitable for performing data mining tasks without having access to the original data. However, the methods proposed so far only target a small to medium number of variables, since their estimates rely on representing all the interdependencies between the variables of the data. High-dimensional data streams, which are becoming more and more frequent due to increasing numbers of interconnected devices, are, therefore, pushing these methods to their limits. To mitigate these limitations, we present an approach that projects the original data stream into a vector space and uses a set of representatives to provide an estimate. Due to the structure of the estimates, it enables the density estimation of higher-dimensional data and approaches the true density with increasing dimensionality of the vector space. Moreover, it is not only designed to estimate homogeneous data, i.e., where all variables are nominal or all variables are numeric, but it can also estimate heterogeneous data. The evaluation is conducted on synthetic and real-world data. The software related to this paper is available at <https://github.com/geilke/mideo>.

1 Introduction

In the context of discrete densities, Geilke *et al.* [6,7] presented online density estimators that not only capture the distribution of data streams but also support data mining tasks. The presented density estimates were described using (ensembles of (weighted)) classifier chains, where each classifier predicts one variable of the stream and is built using the variables of the previous classifiers. This relationship is inspired by the chain rule of densities, according to which the dependencies between the variables are modeled. As long as the density has only a few variables, this method provides an accurate description of the data [6]. But as soon as the dimensionality increases, the number of classifiers and their size grows quickly – making this approach unsuitable for data of high dimensionality. High-dimensional data streams, however, are becoming more and more frequent with the constantly increasing number of interconnected devices that try to measure aspects of their environment to make intelligent decisions. For example, future smart homes may have many sensors measuring various parameters such as temperature or humidity. By learning from past measurements, machine learning algorithms have the possibility of distinguishing between typical and abnormal behavior and can suggest appropriate actions to the user.

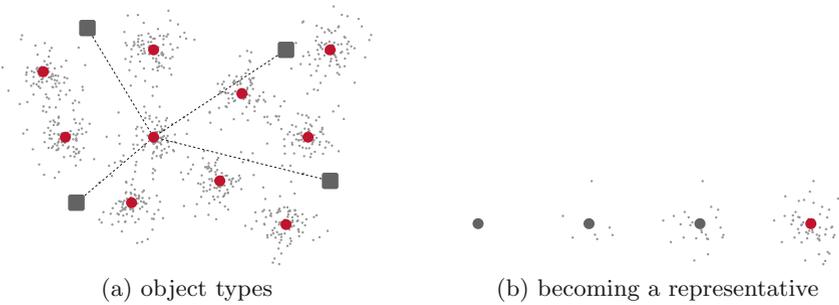


Fig. 1. On the left are the main object types: *landmarks* (dark gray squares), *representatives* (red), and *instances* (gray dots). On the right is the process of becoming a representative: If a distance vector cannot be assigned to a representative or candidate, it is first considered a candidate (big dark gray circle). Over time more instances appear in its neighborhood. If a predefined number is reached, the candidate is turned into a representative (big red circle). (Color figure online)

Whereas an increase of the humidity in the basement can be explained by a tumble dryer that has been started some time ago, such an increase in the bedroom could be due to water entering the room through an open window. The former situation is probably quite normal for a household and requires no action, the latter situation needs attention by the user. Providing an estimate that captures the density of the sensor measurements and provides facilities to perform data mining tasks can be useful to develop such applications.

In this paper, we address the problem of estimating the joint density of heterogeneous data streams with many variables. In data mining, one would usually consider variables in the hundreds or thousands as many variables. For density estimation, however, even 50 binary variables is already considered high-dimensional, as there are 2^{50} value combinations. For each of these combinations, the estimator has to assign a density value, which makes the estimation a challenging task. To perform density estimation on data with many variables, we designed an algorithm, called RED (Representative-based online Estimation of Densities). The main idea is to project the data stream into a vector space of lower dimensionality by computing distances to well-defined reference points. In particular, we distinguish between three types of objects (see Fig. 1 for an illustration): *landmarks*, *representatives*, and *instances*. Landmarks are reference points spanning a vector space for representatives and instances, so that the position of each object can be defined in terms of distances to the landmarks (e.g., the position of the red circle in Fig. 1 that is connected to the four landmarks (red circles) can be specified as a four tuple (d_1, d_2, d_3, d_4) , where d_i is the distance to landmark i , $i \in [1; 4]$). Representatives stand for clusters of instances and will be the main components for estimating probabilities. The landmarks will be used to compute the relative distance of instances, and the representatives will maintain statistical information about instances that have been observed in their neighborhood. To maintain this information, we employ an extended version of

the density estimators proposed by Geilke *et al.* [6]¹, called *EDO* (Estimation of Densities Online), and estimate the distances to nearby instances for each representative. Compared to *EDO*, which directly estimates the density of the instances, *RED* reduces the dimensionality of the dataset to the number of landmarks. If this number is substantially smaller than the number of variables, the model size of the estimators can be substantially reduced, thereby making the approach suitable for data of higher dimensionality.

The main contributions of the paper are:

1. an online density estimator for heterogeneous data with many variables, i.e., data with many nominal and/or numerical variables,
2. a theoretical analysis for the choice of the landmarks,
3. a theoretical analysis for the consistency of the estimates.

2 Related Work

Whereas many data mining tasks have received considerable attention in the context of stream mining recently, only little is known about the estimation of joint densities in an online setting. In this setting, the algorithm has to learn a joint density f solely from the instances of a data stream. Interdependencies between instances are usually not taken into account, which distinguishes it from online learning protocols where the outcome of a given variable is predicted based on past outcomes [3].

Offline density estimation includes recent work based on decision trees [13], where the leaves contain piecewise constant estimators. A similar approach was pursued by Davies and Moore [4] as part of a conditional density estimator. Work towards the estimation of conditional densities has been pursued among others by Frank and Bouckaert [5] and Holmes *et al.* Multivariate densities are frequently estimated using kernel density estimators [8, 14], which is also the predominant direction of the few online variants of density estimation so far. For example, Kristan *et al.* [9–11] proposed a method yielding results that are comparable to the corresponding batch approaches. Xu *et al.* [16] introduced sequences of kernel density estimators to address density estimation from a data stream with only a few variables. The approach presented in this work differs from kernel density estimators in two aspects: (1) the data is projected into a vector space of lower dimensionality by computing distances to reference points, and (2) the basic density estimators are online density estimators that are able to represent complex non-parametric densities.

Datasets with many instances were considered by Peherstorfer *et al.* [12]. They proposed to use a sparse grid where basis functions are not centered around the instances but at grid points. Partitioning the space of data instances was also the strategy pursued by RS-Forest by Wu *et al.* [15], who used a forest of trees

¹ The density estimators are designed for discrete variables, but they can be extended to mixed types of variables by using the conditional density estimators proposed by Eibe and Frank [5]. Details will be provided in a forthcoming journal publication.

to partition the data space. Density estimation on data streams with a greater number of variables has – to the best of our knowledge – not been considered so far, but the methods by Kristan *et al.* [11] and RS-Forest come closest to the requirements and are, therefore, the methods we considered for a comparison.²

Although distances and representative instances have been used before to project data into a space of lower dimensionality (e.g., multidimensional scaling), the approach presented in the paper is different. Whereas other techniques try to preserve the relevant characteristic properties of the data when embedding it into a space of lower dimensionality, RED characterizes the data using landmarks and provides a back translation to the original data. This back translation is a crucial and necessary part to enable density estimation.

The approach pursued by RED is also different from micro-clustering [1]. Whereas micro-clusters maintain simple statistical properties of the data such as the linear sum, RED uses landmarks and Gaussian mixtures to partition the data space and then estimates the full joint density of each partition.

3 Density Estimation Using Representatives

Let X_1, \dots, X_n be a set of variables and let \mathbf{x} be an instance defined over these variables. Given a possibly infinite stream of instances with many variables, we address the problem of estimating its density, $f : X_1 \times \dots \times X_n \rightarrow [0; 1]$, in an online fashion, i.e., only the current instance and its current estimate is provided. In order to determine a density estimate \hat{f} , we propose a method that reduces their dimensionality by using a small set of reference points $L := \{L_1, \dots, L_m\}$, so-called landmarks. With these reference points, it projects the data into a vector space of dimensionality $m < n$ by applying a mapping, $h_L : X_1 \times \dots \times X_n \rightarrow \mathbb{R}^m$, to each instance \mathbf{x} . Here, the i -th component of $h_L(\mathbf{x})$ is defined as the distance between \mathbf{x} and landmark L_i . The estimate \hat{f} of f is then expressed as the product of two independent components: an online estimate \hat{g} that captures the density in the vector space and a *correctionFactor*, which is the expected number of instances that are mapped to the same distance vector – without the correction factor, we would only estimate g . Hence, $\hat{f}(\mathbf{x}) = \hat{g}(h_L(\mathbf{x})) \cdot \text{correctionFactor}$. In the remainder of this section, we give a detailed description of these components and provide a theoretical analysis.

3.1 The Density of the Vector Space

In order to estimate the density g , RED distinguishes three kinds of objects in the vector space: *distance vectors*, *representatives*, and *candidates*. A distance vector is a projected data stream instance, which is determined by computing the distances to the landmarks. A representative is a distance vector together with a density estimator and a covariance matrix, where the density estimator

² Unfortunately, even after several emails, the authors of RS-Forest did not respond to our request to share their program.

Algorithm 1. updateDensityEstimate

Input: landmarks L , instance \mathbf{x} , mapping h_L , number of neighbors $k \in \mathbb{N}$, candidate threshold $\theta_{C \rightarrow R} \in \mathbb{N}$, $C := \{(c, \Sigma, b) \mid \text{candidate } c \text{ covariance, matrix } \Sigma, \text{ recent instances } b\}$, $R := \{(r, \Sigma, e, b) \mid \text{representative } r, \text{ covariance matrix } \Sigma, \text{ estimator } e, \text{ recent instances } b\}$

```

// check representatives
1 Let  $(r, \Sigma, e, b) \in R$  with  $r$  being closest to  $h_L(\mathbf{x})$ 
2  $e.\text{update}(h_L(\mathbf{x}))$  if  $r.\text{isMember}(\mathbf{x}, \Sigma)$ 
// no matching representative?
3 if  $\exists (r, \Sigma, e, b) \in R : r.\text{isMember}(\text{inst}, \Sigma)$  then
    // no matching candidate?
4     if  $\exists (c, \Sigma, b) \in C : c.\text{isMember}(\text{inst}, \Sigma)$  then
        // find  $k$  closest neighbors
5          $pq \leftarrow \text{priorityQueue}()$ 
6         for  $(r, \Sigma, e, b) \in R$  do
7              $nb \leftarrow (r, e, \|r - h_L(\mathbf{x})\|, b)$ 
8              $pq.\text{insert}(nb, \|r - h_L(\mathbf{x})\|)$ 
9          $\text{neighbors} \leftarrow \{pq.\text{peekMin}() \mid k \text{ times}\}$ 
        // compute covariance matrix
10         $\text{sample} \leftarrow \emptyset$ 
11        for  $(r, e, \|r - h_L(\mathbf{x})\|, b) \in \text{neighbors}$  do
12             $\text{sample} \leftarrow \text{sample} \cup \{\mathbf{x}' \mid \mathbf{x}' \in b\}$ 
        // initialize candidate with empty buffer
13         $C.\text{append}((\mathbf{x}, \text{covariance}(\text{sample}), [\mathbf{x}]))$ 
    // matching candidates
14    else
15        for  $(c, \Sigma, b) \in C$  do
16            if  $c.\text{isMember}(h_L(\mathbf{x}), \Sigma)$  then  $b \leftarrow b \cup \{\mathbf{x}\}$ ; break
17        for  $(c, \Sigma, b) \in C$  with  $|b| \geq \theta_{C \rightarrow R}$  do
18             $e \leftarrow \text{initialize EDO estimator}$ 
19             $e.\text{update}(h_L(\mathbf{x}))$ 
20             $R.\text{append}((c, \Sigma, e, b))$ 

```

is supposed to provide a density estimate of nearby distance vectors. Whether a distance is nearby is decided based on its Mahalanobis distance to the representative. A candidate is a precursor of a representative. It will be turned into a representative, if it gathers enough distance vectors around it. The intuition behind these objects is that the landmarks provide a space with certain properties and guarantees, and the representatives and candidates are responsible for modeling the density.

Given the current \hat{g} , the estimate is updated as follows (illustrated by Fig. 1): The next instance \mathbf{x} is first projected into the vector space by applying h_L . Then the resulting distance vector \mathbf{v} is tested against all representatives. If a representative is found, \mathbf{v} is forwarded to the corresponding density estimator. Otherwise, \mathbf{v} is tested against all candidates. If a candidate is found, \mathbf{v} is assigned to the candidate. Otherwise, \mathbf{v} becomes a candidate itself. Algorithm 1 describes

the update procedure of RED in more detail. In lines 1–2, the EDO estimates are updated if the new instance is considered as a member of that representative. If it does not belong to any representative (line 5), we distinguish two cases: the instance does not belong to any existing candidate (lines 3–16) or it does (line 17–26). In the first case, the given instance becomes a candidate itself, as no matching representative nor candidate has been found. For this purpose, we determine the k closest representatives and compute a covariance matrix from their most recent samples. The result is a new covariance matrix, which will be used in the future to decide whether or not an instance belongs to the new candidate. In the second case (there are candidates to which the instance belongs), there is no need to create a new candidate. Instead, the buffers of matching candidates are simply updated. However, since the recent updates could result in buffer sizes of $\theta_{C \rightarrow R}$ instances, the update procedure finishes with checking the buffer size of all candidates (lines 21–25) and turning candidates to representatives that have more than $\theta_{C \rightarrow R}$ instances.

Membership Tests. Whether an instance belongs to a candidate or representative is decided by employing a multivariate normal density $\mathcal{N}(\mathbf{v}; \Sigma)$, where \mathbf{v} is the distance vector of the representative instance and Σ is a covariance matrix computed from instances in the neighborhood – when a new candidate is created, it is computed from recent samples of neighboring representatives. The membership decision is based on the Mahalanobis distance, which is computed as follows: $\sqrt{(\mathbf{x} - \mathbf{r})^T \Sigma^{-1} (\mathbf{x} - \mathbf{r})}$. Any vector with a Mahalanobis distance less than a user-defined threshold is then considered a member.

Handling of Noise. Almost all real-world applications suffer from certain degrees of noise. Hence, handling noise is of paramount importance for density estimators but in many cases difficult. In an online setting, the problem is even more severe, since future instances cannot be included into the decision making process. The EDO estimators employed by RED are able to handle noise, but in order to keep them as clean as possible, it is important that a noisy instance does not become a representative in the first place. Otherwise, this instance and every instance in its neighborhood becomes inevitably a part of the estimate. Therefore, RED distinguishes between candidates and representatives. If an instance cannot be assigned to an existing representative, it is first considered a candidate for becoming a representative. Only if enough instances are gathered around the candidate, it becomes an actual representative.

Concept Drift. In real-world applications, the distribution of data streams is changing constantly and a density estimator has to adapt to these changes to provide reliable estimates. In order to address this problem, RED pursues a timestamp-based solution. However, old instances are not simply discarded when they become too old, but candidates and representatives are discarded if no instance has been assigned to them within a certain period of time. This time period is specified as a parameter and can be adjusted according to the

smallest probability values that should be covered by the estimate – using Chernoff bounds, the parameter can be computed with high confidence.

When setting this parameter, one should also consider the parameter $\theta_{C \rightarrow R}$, as a high value for $\theta_{C \rightarrow R}$ prevents rare instances from becoming a part of the density estimate. In our experiments, we usually set $\theta_{C \rightarrow R}$ to 100, which is large enough for a statistical test but not too large to exclude less frequent instances.

3.2 Distance Measure

With landmarks, high-dimensional data can be mapped to a lower dimensional vector space. But dependent on the number and the choice of the landmarks, the resulting vector space could still be relatively large, so that the distance measure has to be chosen with care. For high-dimensional spaces, the Manhattan distance (1-norm) or a fractional distance measure is usually the best choice [2], so that we prefer p -norms with small p . Employing a p -norm, the mapping $h_L : X_1 \times \dots \times X_n \rightarrow V_1 \times \dots \times V_m$ with $V_j \subseteq \mathbb{R}$, $1 \leq j \leq m$, is defined as

$$h_L(\mathbf{x})[V_i] := \left(\sum_{X_j \in X} \|l_i[X_j] - \mathbf{x}[X_j]\|^p \right)^{\frac{1}{p}},$$

where $\|\cdot\|$ computes the distance for the given variable values and is defined as the difference $\frac{l_i[X_j] - \mathbf{x}[X_j]}{\max(X_j) - \min(X_j)}$ for numeric values and $\frac{l_i[X_j] - \mathbf{x}[X_j]}{\#values} \in [0; 1]$ for nominal values. For p , we select values from the range $(0; 2]$, which corresponds to the Euclidean distance for $p = 2$, the Manhattan distance for $p = 1$, and to fractional norms for $0 < p < 1$. The denominator $\max(X_j) - \min(X_j)$ can only be estimated, as the currently observed minimum and maximum values cannot be determined with certainty in a streaming setting, making a correct normalization impossible. For typical applications, however, an estimate is probably more than sufficient, since extreme deviations are most likely due to a concept drift.

3.3 Choice of the Landmarks

If the data stream is projected into a vector space with lower dimensionality, information about the original instances will possibly be lost. In particular, some of the variable interdependencies are no longer visible, as the mapping h_L only adds up the distances of individual variables. As a consequence, instances may be projected into the same point of the vector space, i.e., there are \mathbf{x} and \mathbf{x}' , such that $h_L(\mathbf{x}) = h_L(\mathbf{x}')$ but $\mathbf{x} \neq \mathbf{x}'$. If the original instances has only nominal variables ($\{X_1, \dots, X_k\}$) and each variable has $|X_i|$, $1 \leq i \leq k$, many values, then there are already $\prod_{i=1}^k 2 \cdot (|X_i| - 1)$ many possible instances that are mapped to the same distance value. RED would treat all of these instances equally when computing their density value, which poses no problem to the density estimate, as long as similar distances to the landmarks correspond to similar density values of the instances. But it implies that the information encoded by the distances

to the landmarks needs to be sufficiently good. Therefore, we propose to choose the landmarks in such a way that the estimate approaches the accuracy of the underlying density estimator as $|L|$ approaches n :

Definition 1. Let $X := \{X_1, \dots, X_n\}$ be the set of variables, and let m be the requested number of landmarks. Then the first landmark L_1 is defined as $(0)_{1 \leq j \leq n}$ and landmark L_{i+1} , $1 \leq i < m$, is defined as

$$\left((i - j + 1) \cdot \frac{\max(X_j) - \min(X_j)}{m} \right)_{1 \leq j < i} \circ (1) \circ (0)_{j > i}$$

where $\max(X_j)$ and $\min(X_j)$ are the currently observed maximum and minimum, respectively. The set of landmarks is denoted by $L := \{L_i \mid 1 \leq i \leq m\}$.

By construction of h_L and by the choice of the landmarks, the mapping h_L projects any two instances \mathbf{x} and \mathbf{x}' to different points in the vector space as long as $\mathbf{x} \neq \mathbf{x}'$, $|L| = n + 1$, and certain assumptions hold. (An example is given below.) Hence, the mapping becomes injective:

Lemma 1. If $|L| = n + 1$, landmark L_i , $i \in [1; n + 1]$, is defined as in Definition 1, and \max is the actual maximum for all $X_j \in X$, then the projection mapping $h_L : X_1 \times \dots \times X_n \rightarrow V_1 \times \dots \times V_m$ with $V_j = \mathbb{R}$, $1 \leq j \leq m$, is injective.

Proof. Under the assumption that $\max(X_j)$ is the actual maximum for all $X_j \in X$, $L_j[X_j]$ is always larger than $h_L(\mathbf{x})[V_j]$. Hence, for $x_{j_1} \neq x_{j_2} \in X_j$, $h_L(\mathbf{x}_{j_1})[V_j]$ is not equal to $h_L(\mathbf{x}_{j_2})[V_j]$. (Please notice that this would not have been the case, if we had chosen $L_j[X_j]$ to be $\frac{1}{2} \cdot \max(X_j)$, since h_L does not consider the sign of differences, e.g., $(\min(X_j) + 0.2) - \frac{1}{2} \cdot \max(X_j)$ and $(\max(X_j) - 0.2) + \frac{1}{2} \cdot \max(X_j)$ result in the same distance).

So individual variable values do not cause two different instances to have the same distance vector. It remains to show that this property is preserved when computing the summation over all variable differences in h_L . For this purpose, we project $X_1 \times \dots \times X_n$ into \mathbb{R}^n and first show that, if $|L| = n + 1$, all vectors \mathbf{x} that start in the origin of \mathbb{R}^n and are mapped to the same distance vector have the same length. Let $h_L(\mathbf{x}) = (v_1, v_2, \dots, v_m)$ be the distance vector of \mathbf{x} . We can determine the possible lengths of all instance vectors \mathbf{x} that are mapped to $h_L(\mathbf{x})$ by finding the solution for the following system of equations:

$$\begin{pmatrix} 1 & 0 & 0 & \dots & 0 & v_1 \\ 1 \cdot s_1 & 1 & 0 & \dots & 0 & v_2 \\ 2 \cdot s_1 & 1 \cdot s_2 & 1 & \dots & 0 & v_3 \\ & & & \vdots & & \\ n \cdot s_1 & (n-1) \cdot s_2 & (n-2) \cdot s_3 & \dots & 1 & v_n \end{pmatrix},$$

where s_j equals $\frac{\max(X_j) - \min(X_j)}{m}$. Due to the choice of the landmarks, the left-hand side is a squared matrix with rank n . Hence, the system of equations has only one unique solution: $\mathbf{x} = A^{-1} \mathbf{b}$.

Dependent on the norm employed by h_L , there are fewer or more instances having the same length in \mathbb{R}^n . In case of the Euclidean norm, for example, the corresponding vectors having the same distances to the landmarks L_2, \dots, L_{n+1} lie on the border of a $(n-1)$ -dimensional norm sphere (notice that L_1 is excluded here). In order to ensure that the mapping h_L is injective, we simply have to include the landmark L_1 , which introduces another dimension and reduces the border of the $(n-1)$ -dimensional norm sphere to a single point. The same approach is also valid for arbitrary p -norms, which we prove by constructing a contradiction: Let L be defined as in Definition 1. Assume that there are $\mathbf{x} \neq \mathbf{y}$ in the projected vector space, such that $\forall L_i \in L : \left(\sum_{X_j \in X} \|L_i[X_j] - x[X_j]\|^p \right)^{\frac{1}{p}} = \left(\sum_{X_j \in X} \|L_i[X_j] - y[X_j]\|^p \right)^{\frac{1}{p}}$. Due to the projection of \mathbf{x} and \mathbf{y} into \mathbb{R}^n and due to the definition of $\|\cdot\|$, $\|\cdot\|$ becomes $|\cdot|$ in \mathbb{R}^n . From the first part of the proof, we can conclude that for all $L_i \in L$ and for all $X_j \in X$:

$$\begin{aligned} |L_i[X_j] - x[X_j]|^p &= |L_i[X_j] - y[X_j]|^p \\ \Leftrightarrow |L_i[X_j] - x[X_j]| &= |L_i[X_j] - y[X_j]|, \end{aligned}$$

As this equation also has to hold for L_1 ($L_1 = (0)_{1 \leq i \leq n} \in L$) and as for all $X_j \in X : x[X_j], y[X_j] \geq 0$, this implies that for all $X_j \in X : x[X_j] = y[X_j]$, which contradicts the assumption that $\mathbf{x} \neq \mathbf{y}$. \square

This theorem makes a valuable statement about the validity of the method: Although $n+1$ landmarks is probably infeasible for extremely high-dimensional data streams, we know that additional landmarks are beneficial for the accuracy of the method. Hence, it is up to the user whether the accuracy or memory consumption is more critical.

3.4 Correction Factor

If $|L| < n+1$, the projection mapping h_L maps several instances from the original data stream into the same point of the vector space. When RED estimates the density value of that point, it has actually estimated the density value of all instances that are mapped to it. Since we have no additional information on how to divide the density value among those points, we will divide it equally.

To obtain the density value from the original data stream, we have to multiply it by the integral $\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \dots \int_{-\infty}^{+\infty} (x_i, x_{i+1}, \dots, x_p) dx_{i+1} dx_{i+2} \dots dx_n$, which is the expected density value of the variables X_i, X_{i+1}, \dots, X_n . In case where all variables are discrete, this would simply be $\frac{1}{\prod_{i=1}^k |X_i|}$. For a continuous variable X_j , we will approximate the integral using a sampling technique. In particular, we employ a conditional density estimator for $f(X_j | V_1, V_2, \dots, V_m)$ and sum $f(x_j | v_1, v_2, \dots, v_m)$ over $\min(X_j) \leq x_j \leq \max(X_j)$ with step size $\frac{\max(X_j) - \min(X_j)}{|S|}$ and sample size $|S|$. Hence, the correction factor is 1, if $|L| \geq n+1$.

3.5 Illustrative Example

To illustrate the density estimation RED, we give a small example: We generated a synthetic data stream of dimensionality 3, for which we selected the landmarks $L_1 = (1, 0, 0)$, $L_2 = (0.2, 1, 0)$, and $L_3 = (0.3, 0.2, 1)$ and projected it into a vector space of dimensionality 3. Subsequently, we also projected the data into a vector space of dimensionality 2 – this time with the landmarks $L_1 = (1, 0, 0)$ and $L_2 = (0.2, 1, 0)$. As Fig. 2 illustrates, the instance clusters are still visible after applying h_L to the data (see (a) and (d)). The relative positions remain roughly the same, but due to the landmarks, they capture a different area in the vector space. For the vector space that is induced by two landmarks, the instance clusters are arranged similarly. When the instances have been mapped to the vector space, it is up to the representatives to model the density. The density values of the original data stream can then be computed by retranslating the density value using the *correctionFactor*, where the *correctionFactor* accounts for the instances that have been mapped to the same point in the vector space.

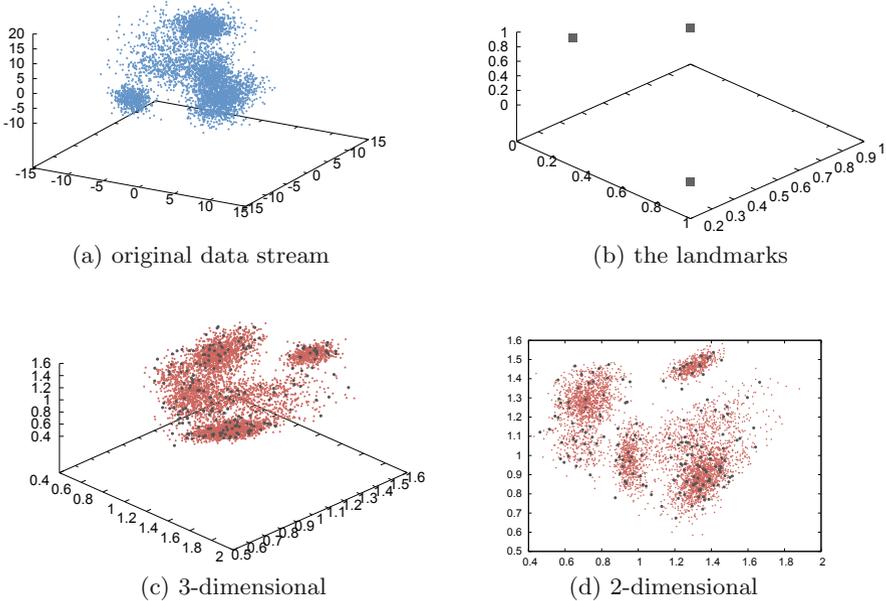


Fig. 2. Illustrates a synthetic data stream (see Plot (a)) that is projected with three landmarks (see Plot (b)) into a vector space of dimensionality 3 (see Plot (c)) and with two landmarks to dimensionality 2 (see Plot (d)). Please notice the dark gray dots in Plot (c) and (d) are the representatives. Due to the Mahalanobis distance, dense regions have more representatives than sparse regions, which helps to model the density more accurately. (Color figure online)

3.6 Consistency

The consistency of a density estimator is a desirable property, as it ensures that the estimator approaches the true density. We have already shown that the number of instances that are mapped to the same point in the vector space can be controlled by the number of landmarks. In the following, we will prove that RED estimates are consistent, if some further conditions hold:

Theorem 1. *If $|L| = n + 1$, L is defined as in Definition 1, \min is the actual minimum for all $X_j \in X$, \max is the actual maximum for all $X_j \in X$, and \hat{g} is consistent, then \hat{f} is consistent.*

Proof. If $|L|$ equals $n+1$, then the correction factor is 1, so that it remains to show that $\hat{f}(\mathbf{x}) = \hat{g}(h_L(\mathbf{x}))$. Furthermore, as Algorithm 1 partitions the vector space into independent subregions where each subregion has its own online density estimator, it suffices to show that the estimator of each subregion is consistent.

Since $|L|$ equals $n + 1$ and the assumptions for \min and \max hold, h_L is injective according to Lemma 1. Hence, for any two instances \mathbf{x}_1 and \mathbf{x}_2 that only differ in variable X_j by a small amount, it follows that

$$h_L(\mathbf{x}_1[V_i])^p - h_L(\mathbf{x}_2[V_i])^p = \sum_{k=1}^p c_k \cdot \|\mathbf{x}_1 - \mathbf{x}_2\|,$$

by definition of h_L , where $1 \leq i \leq n + 1$ and c_k are constants that have two factors: l_i and a constant that results from the binomial theorem. In other words, the density in the vector space is only shifted and compressed, but the original information of f is completely contained in g . Therefore, we can conclude that $\hat{f}(\mathbf{x}) = \hat{g}(h_L(\mathbf{x}))$. \square

4 Evaluation

In this section, we analyze the behavior of RED³ with respect to its parameters on synthetic datasets and evaluate its capability to estimate joint densities on several real-world datasets. RED has several parameters that have to be chosen: the number of landmarks $|L|$, the Mahalanobis distance, the threshold of becoming a representative $\theta_{C \rightarrow R}$, and the distance measure p . As $\theta_{C \rightarrow R}$ is mostly relevant for drift detection or for application tasks such as outlier detection, we do not discuss this parameter here. Also not discussed is the parameter p , as a detailed analysis would be in conflict with the given space constraints. Hence, we focus our experimental analysis of the parameters on the number of landmarks $|L|$ and the Mahalanobis distance. For $|L|$, we consider the values 2, 3, 5, 10, and 20. For the Mahalanobis distance, we consider the values 0.1, 0.5, 1.0, 2.0, 5.0, and 10.0.

³ An implementation of RED is available as part of the MiDEO framework: <https://github.com/geilke/mideo>.

As datasets, we generated synthetic data consisting of 1, 2, 5, or 10 multivariate Gaussians in a d -dimensional vector space with $d \in \{2, 3, 5, 10, 20\}$. The mean variables and covariance matrices were drawn independently and uniformly at random, where the values for the mean have been drawn from the interval $[-10; 10]$ and the values for the covariance matrix from the interval $[0.5; 3]$.

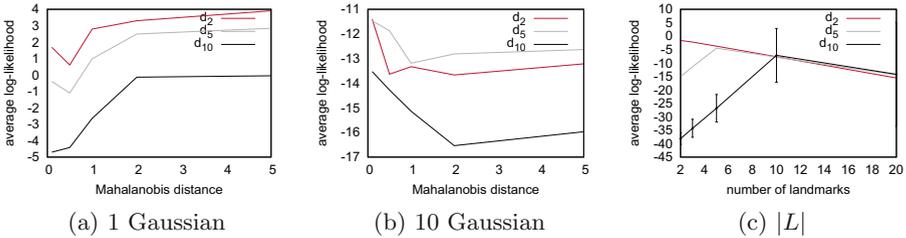


Fig. 3. For data of different dimensionality, the figures give some details about the behavior of RED when the Mahalanobis or the number of landmarks is increased. All plots are aggregated over all synthetic datasets. (Color figure online)

The influence of the number of landmarks is illustrated by Fig. 3. The shapes of the curves show that the performance is increasing with the number of landmarks until it reached the dimensionality of the dataset. So, for d_2 , the peak performance is reached at 2, for d_5 , the peak performance is reached at 5, and, for d_{10} , the peak performance is reached at 10. The increase for $|L| \leq d$ is completely in line with Theorem 1. The decrease for $|L| > d$ can be explained by the increase of the vector space: Due to higher dimensionality of the vector space, fewer and fewer instances share the same space and, hence, fewer instances are available to provide an estimate for this region. This effect is also responsible for the increase of the variance for increasing numbers of landmarks (as visible for d_{10}). Due to the small number of instances per region, the density estimators are more sensitive to smaller changes, which results in an increased variance. So generally, up to d landmarks are beneficial for the performance, but the closer $|L|$ gets to d , the more instances are required to compensate for the higher variance.

The effect of the Mahalanobis distance is summarized by Figs. 3 and 4. For lower dimensional datasets (e.g., d_2 and d_3), the performance is slightly degrading for minor increases of the Mahalanobis distance. For $M_{5,0}$ and $M_{10,0}$, however, we already see substantial improvements, which can be explained by fewer numbers of representatives, which have more instances at their disposal to provide a good estimate. For higher dimensional datasets (e.g., d_{10} and d_{20}), this trend is reverted, and we consistently observe that a low Mahalanobis distance is the better choice among the given selection. This can be explained by the possibility for each representative to specialize on regions with many instances. Otherwise, one representative would be responsible for a diverse set of instances. This observation is further supported by Fig. 3. If there is only one Gaussian, a

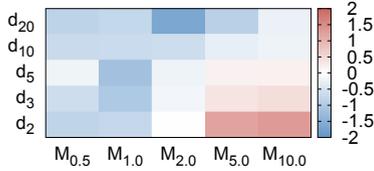


Fig. 4. The heatmap summarizes the effects of the Mahalanobis distance for data of different dimensionality d_i ($i \in \{2, 3, 5, 10, 20\}$). It shows the improvement (red) and degradation (blue) if RED uses a specific Mahalanobis distance ($M_{0.5}, M_{1.0}, M_{2.0}, M_{5.0}, M_{10.0}$) compared with a Mahalanobis distance of 0.1. (Color figure online)

higher Mahalanobis distance is beneficial. But if we have several Gaussians (e.g., 10), a larger Mahalanobis distance leads to an degradation of the performance. So generally, one can say the higher the dimensionality of the data, the lower should be the Mahalanobis distance.

If data mining and machine learning should be performed on RED estimates, the estimates have to describe the density of the data as accurately as possible. Since several compromises have been made to enable the estimation of data with many variables, we do not expect to outperform other density estimators. However, the performance should be in the same order of magnitude.

In order to evaluate RED on real-world data, we selected the state-of-the-art online density estimation method, which is the online kernel density estimator oKDE by Kristan *et al.* [11], and compared the performance on four publicly available datasets: covertypes (581,012 instances, 54 attributes), electricity (45,313 instances, 9 attributes), letter (19,999 instances, 17 attributes), and shuttle (58,000 instances, 10 attributes). For every parameter setting and for every dataset, the average log-likelihood is computed 15 times. In order to take possible concept drifts into account, the log-likelihood was computed in a prequential way, i.e., the log-likelihood of a given instance has been computed before using it for training. The instances used to compute the initial estimator (the first 100) were excluded from this computation.

The results are summarized in Fig. 5. The most apparent observation is the performance increase with increasing numbers of landmarks. As already observed on synthetic data, this performance increase is accompanied with an increase of the variance. Dependent on the dataset the effect is visible to different degrees and does not even depend on the number of variables (electricity vs. shuttle). But most surprising is probably the abrupt increase on the covertypes dataset. A more detailed analysis of this matter revealed that this is due to the nature of its instances. Covertypes has 10 numerical variables, 43 binary variables, and one further nominal variable. As most of the binary variables are 0 for almost all instances, they do not provide sufficient additional information to justify more landmarks. Hence, with every new landmark, the estimators have fewer instances and provide estimates that are more sensitive to individual instances.

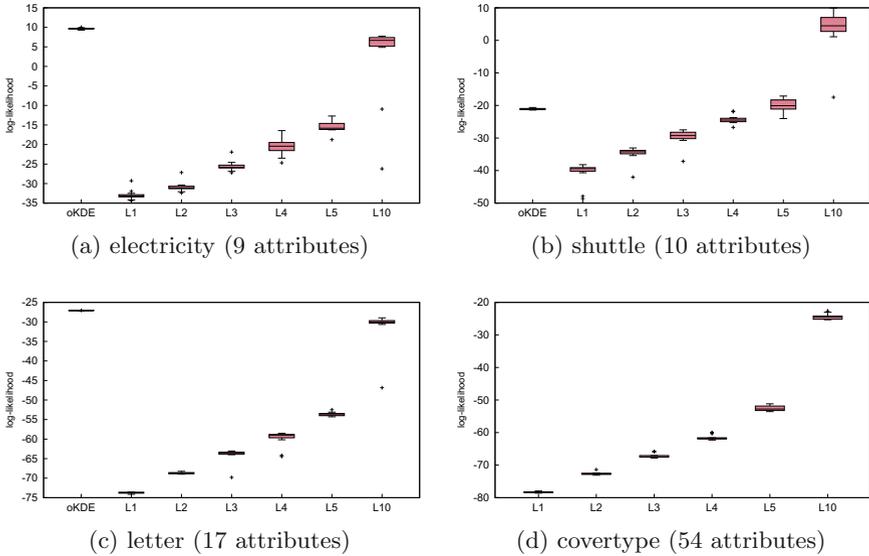


Fig. 5. The figure shows a comparison of oKDE with RED for varying numbers of landmarks ($|L| \in \{1, 2, 3, 4, 5, 10\}$) in the case of RED. On the y-axis is the average log-likelihood computed in a prequential way. (Color figure online)

When we compare RED to oKDE, we observe that RED performs surprisingly well. For electricity and letter, RED is approaching the performance of oKDE. For shuttle, the performance is even better than that of oKDE, if RED uses 5 or 10 landmarks. For covertime, oKDE was not even able to process the dataset within 15 hours, whereas RED was able to produce results for all landmark sizes. Hence, RED is able to compete with oKDE on low dimensional data, when a sufficient number of landmarks is chosen, while it can also handle high-dimensional data (e.g., more than 50 attributes). How many landmarks are sufficient for a specific datasets depends on two aspects: its intrinsic dimension and the selected landmarks (because the landmarks determine which dimensions are considered for evaluating the distance function).

Considering that we made several compromises to enable the density estimation of data streams with many variables, RED performed very good on low- and medium-sized data streams. The insights we gained about the parameters should be a useful guide for applications to other data streams. Alternatively, one could also follow a multi-layer approach where three or four RED estimators are initialized with different parameter settings. When enough instances are available, one could then choose the estimator with highest log-likelihood. This is enabled by excellent runtime behavior.

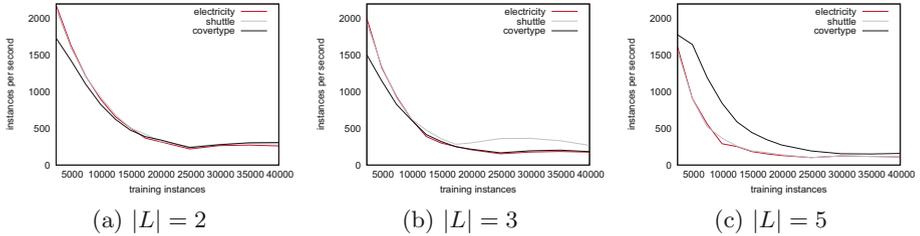


Fig. 6. Number of instances processed per second. (Color figure online)

4.1 Runtime

RED offers many opportunities for parallelism. For our evaluation, however, we wanted to keep the implementation simple and avoided any advanced optimizations. The current implementation is still fast enough for data streams applications, as Fig. 6 summarizes. The general behavior is the same for all tested datasets (electricity, shuttle, and covertype). In the beginning, when almost no representatives are discovered and the density estimators are still very simple, the RED estimate is able to process 1400 or more instances per second. Then, with increasing numbers of training instances, it drops to several hundred instances per second before it stabilizes (at 100 to 300 instances per second, depending on the number of landmarks). This is in line with the expected behavior of the method. First, the instances are required to find a partitioning of the vector space. When this partitioning is converging and the corresponding density estimators have received a larger number of instances, the processing speed of the estimator becomes more and more constant.

5 Conclusions

We proposed a new approach for estimating densities of heterogeneous data streams with many variables, which reduces the dimensionality of the data by projecting it into a vector space. In particular, the algorithm chooses a small number of instances, called landmarks, and creates a vector space in which the position of each instance is computed in terms of distances to the landmarks. Subsequently, the density is described by partitioning the vector space with representatives and estimating the density of each partition by employing online density estimators. In the theoretical analysis, we showed the validity and consistency of the presented density estimator. With experiments on synthetic and real-world data, we showed that – despite the compromises that had to be made to enable the estimation of densities with many variables – RED produces estimates having a comparable performance to that of state-of-the-art density estimates. Keeping in mind that other approaches are possibly not able to handle large numbers of variables, RED could be, at this point, the only available option for some data streams.

In the future, we plan to further analyze the choice of landmarks and plan to perform data mining tasks such as outlier detection, the detection of emerging trends and inference on RED estimates.

References

1. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: Proceedings of the 29th International Conference on Very Large Data Bases (VLDB), pp. 81–92 (2003)
2. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the surprising behavior of distance metrics in high dimensional spaces. In: Proceedings of the 8th International Conference on Database Theory, pp. 420–434 (2001)
3. Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press (2006)
4. Davies, S., Moore, A.W.: Interpolating conditional density trees. In: Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence, pp. 119–127 (2002)
5. Frank, E., Bouckaert, R.R.: Conditional density estimation with class probability estimators. In: Proceedings of the First Asian Conference on Machine Learning, pp. 65–81 (2009)
6. Geilke, M., Karwath, A., Frank, E., Kramer, S.: Online estimation of discrete densities. In: Proceedings of the 13th IEEE International Conference on Data Mining, pp. 191–200 (2013)
7. Geilke, M., Karwath, A., Kramer, S.: A probabilistic condensed representation of data for stream mining. In: Proceedings of the 1st International Conference on Data Science and Advanced Analytics, pp. 297–303 (2014)
8. Hwang, J.N., Lay, S.R., Lippman, A.: Nonparametric multivariate density estimation: a comparative study. *IEEE Trans. Signal Process.* **42**(10), 2795–2810 (1994)
9. Kim, J., Scott, C.D.: Robust kernel density estimation. *J. Mach. Learn. Res.* **13**, 2529–2565 (2012)
10. Kristan, M., Leonardis, A.: Online discriminative kernel density estimation. In: 20th International Conference on Pattern Recognition, pp. 581–584 (2010)
11. Kristan, M., Leonardis, A., Skocaj, D.: Multivariate online kernel density estimation with Gaussian kernels. *Pattern Recogn.* **44**(10–11), 2630–2642 (2011)
12. Peherstorfer, B., Pflüger, D., Bungartz, H.: Density estimation with adaptive sparse grids for large data sets. In: Proceedings of the 2014 SIAM International Conference on Data Mining, pp. 443–451 (2014)
13. Ram, P., Gray, A.G.: Density estimation trees. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 627–635 (2011)
14. Scott, D.W., Sain, S.R.: *Multi-Dimensional Density Estimation*, pp. 229–263. Elsevier, Amsterdam (2004)
15. Wu, K., Zhang, K., Fan, W., Edwards, A., Yu, P.S.: Rs-forest: a rapid density estimator for streaming anomaly detection. In: Proceedings of the 14th International Conference on Data Mining, pp. 600–609 (2014)
16. Xu, M., Ishibuchi, H., Gu, X., Wang, S.: Dm-KDE: dynamical kernel density estimation by sequences of KDE estimators with fixed number of components over data streams. *Front. Comput. Sci.* **8**(4), 563–580 (2014)