Anomaly detection with kernel density estimation on manifolds

1 Introduction

```
Background (anomaly detection with density) with summarized contribution

    Anomaly detection on very high-dimensional data. Applications. Main contribution.

Anomaly detection using densities, other methods for anomaly detection. Based manifold lea

Density estimation methods, KDE with distortion, 2-D variable kernel denstiy estimate with

- Main contributions: better anomaly detection

    fix distortion with Riemmannian matrix

    anomaly detection using density estimates

    kernel density estimate is not accurate

    improve density estimation with distortion in the embedding plot
```

The rest of the paper is organized as follows. In Section 2, we present the proposed algorithm to detect anomalies based on variable kernel density estimates of manifold embeddings. In this section, we provide justification for the use of Riemannien matrix as the bandwidth of variable kernel density estimation, including the comparison with fixed bandwidth. Section 3 is composed with two simulations with the proposed algorithm; the first deals with a 2-dimensional meta data and the second with a 100-dimensional meta data. Section 4 contains the application to visualize and identify anomalies in the [TODO] data. Conclusions and discussions are presented in Section 5.

2 Variable kernel density estimation with manifold embeddings

```
Idea: whole process for the main contribution
```

```
Kernel density estimate with variable bandwidth
```

2.1 Notations

2.2 Two dimensional kernel density estimation

In general a multivariate kernel density estimate looks something like this

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{N} K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i)$$

where if a Gaussian kernel is used

$$K_{\mathbf{H}}(\mathbf{x} - \mathbf{x}_i) = (2\pi)^{-d/2} |\mathbf{H}|^{-1/2} \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{x}_i)' \mathbf{H}^{-1} (\mathbf{x} - \mathbf{x}_i)\right]$$

The matrix $\mathbf{H}$ is called the bandwidth matrix and is very important. You will often read that the bandwidth matrix is about smoothing and it is. But an alternative way to think of kernel estimation is that kernel densities 'borrow strength' from nearby points and the bandwidth determines what is "nearby". If the bandwidth is large then all points are "nearby" and we get an overly smooth kernel density estimate. The interesting thing about a bandwidth matrix is that it allows for different notions of what is "nearby" along different coordinates and even along diagonal directions.

```
Riemannian matrix could be used to measure distortion
```

```
Use riemannian matrix as variable bandwidth
    Bandwidth is a symmetric positive-definite square matrix, similar to riemannian matrix
```

2.3 Use riemannian matrix as variable bandwidth

The Riemannian estimated using the method of Perrault Joncas and Meila gives some idea of the distortion of an embedding (or so they claim). Mapping the points through a non-linear function "stretchs" some regions of space and "shrinks" others. The Riemannian gives us an idea of the direction and angle of this stretching. The Riemannian is quite a technical concept but an important thing to understand is that the estimate that comes out of Perrault Joncas algorithm is a square matrix.

We saw how points that are far apart in the embedding may not have been so far apart on the original manifold. The Riemannian gives us some way of correcting this. Similarly a bandwidth matrix in a kernel density estimate is all about determining the "directions" in which there should be more or less "closeness". So the basic idea is to replace the kernel density estimate with

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{N} K_{\mathbf{H}_i}(\mathbf{x} - \mathbf{x}_i) K_{\mathbf{H}_i}(\mathbf{x} - \mathbf{x}_i) = (2\pi)^{-d/2} |\mathbf{H}_i|^{-1/2} \exp\left[-\frac{1}{2} (\mathbf{x} - \mathbf{x}_i)' \mathbf{H}_i^{-1} (\mathbf{x} - \mathbf{x}_i) \right]$$

where H_i is either the Riemannian or the inverse of the Riemannian (I am not totally sure which one). Notice that the bandwidth matrix is different for each point. This makes it a kernel density estimate with local smoothing, which is quite interesting, but we should take care to understand the properties of such things.

```
- Manifold learning distortion


    dimension reduction using manifold learning
    Riemannian matrix to measure topological distortion
    distortion is informative for true embedding
```

The aim of the project is not to evaluate different manifold learning algorithms. Although ISOMAP worked OK here it is still distorted and in real examples you will never 'know' that the data are uniform on the sphere. Instead the idea of this project is to do kernel density estimation in a way that takes distortion into account.

3 Simulation

In this section, we exam two scenarios for both low and high dimensions to test our proposed algorithm. For visualization purpose, Section 3.1 presents a 2-D meta data example. We first simulate the data of size $N = 2000$ from a mixture of four Gaussian kernels with the same covariance but different means, each consisting of 500 points. Different mapping functions are then applied to the 2-D meta data so that they now lie on a 3-D feature space, which gives the input data for different manifold learning algorithms, including ISOMAP, LLE, Laplacian Eigenmaps, t-SNE, and UMAP. The embedded dimension is set as $D = 2$, same as the meta data dimension. This enables us to compare the manifold learning embedding with the true meta data. We could now apply the algorithm described in [TODO]. Similarly, the second simulation in Section 3.2 is based on a 5-D meta data embedded in a 100-D space and the corresponding embedding dimension is $D = 5$.

3.1 2-D Meta data from a Gaussian Mixture Model

We first generate a 2-dimensional data of size $N = 2000$ from a Gaussian mixture model with four components with means $\boldsymbol{\mu_1} = (0.25, 0.25)', \boldsymbol{\mu_2} = (0.25, 0.75), \boldsymbol{\mu_3} = (0.75, 0.25), \boldsymbol{\mu_4}(0.75, 0.75)$ and the same variance-covariance matrix $\boldsymbol{\Sigma}_i = diag(0.02, 0.02), i = 1, 2, 3, 4$. The mixture proportions

are equally set as $\pi_i = 0.25, i = 1, 2, 3, 4$. Then the mixture Gaussian mixture density function is a weighted linear combination of the four component Gaussian densities as

$$P(\boldsymbol{X} = \boldsymbol{x}) = \sum_{i=1}^{4} \pi_i \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_i|^{-1}} \exp\{-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu_i})'\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x} - \boldsymbol{\mu_i})\}. \quad (1)$$

Figure 1 shows the 2-dimensional meta data and the colors indicate the true density of all data points, with brighter colors showing high densities and dark colors showing low densities. We then define outliers as points with lowest densities shown in black and typical points with highest densities shown in yellow. Based on the true density plot, the outliers are scattered in the middle and the outer area of the whole structure, while typical points are near the means of four kernels. This is the baseline to be compared with outliers from variable kernel density estimates.

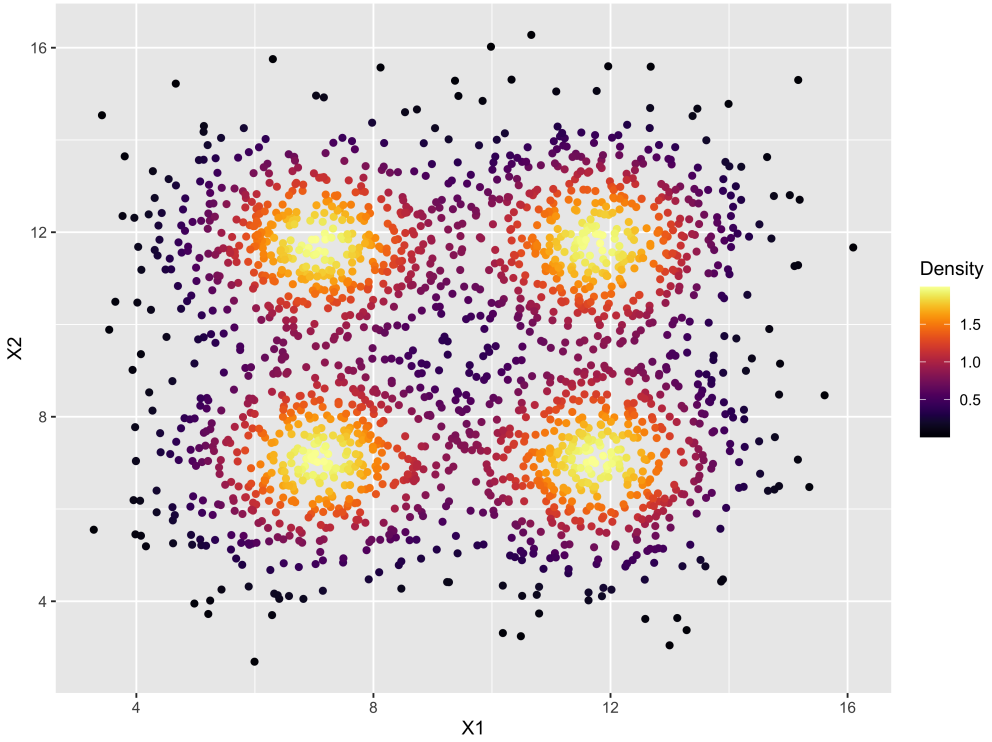

Figure 1: *True density of the Gaussian mixture model of four kenels with means* $(0.25, 0.25), (0.25, 0.75), (0.75, 0.25), (0.75, 0.75)$ *and the same variance-covariance matrix* $diag(0.02, 0.02)$

Given the 2-D meta data, multiple mapping functions could be applied to embed the data in a 3-D space. One of the most famous example in manifold learning is the swiss roll data, with the mapping function as below.

$$\begin{cases} X = X_1 \times \cos X_1, \\ Y = X_2, \\ Z = X_1 \times \sin X_1. \end{cases}$$

Now we are able to apply different manifold learning algorithms to reduce the dimension back to 2, and further estimate the density of the embedding to detect anomalies. According to the density estimates, we could rank the data points and then identify which observations lie in an highest density region of specified coverage, eg. 1%, 50%, 99%, >99%. The top outliers with lowest densities are also indexed in blue, which we call *true outliers*. For each of the four manifold learning methods, namely ISOMAP, LLE, t-SNE, and UMAP, Figure 2 presents in one column the 2-D embedding plot, with the colors indicating the densities, the top row for true densities from the Gaussian mixture model, the middle row for highest density region plots with densities from our proposed variable KDE, and the bottom for similar HDR plots with densities from KDE with fixed bandwidth. Since there are four kernels in the meta data and darker points are the defined outliers, we could derive that for ISOMAP and LLE, variable kernel density estimates could detect the number of kernels more accurately than the KDE with fixed bandwidth. As for t-SNE and UMAP, both embeddings are quite close to each other with points clustered together, but the mixture of kernel structure is not as clear as ISOMAP and UMAP. Instead, we could roughly compare the location of outliers. For t-SNE, the middle plot have indexed more true outliers than the bottom plot; while for UMAP, both the middle and bottom plot fail to find the true outliers, but the outliers from fixed bandwidth are more scattered along the curved embedding.

For comparison, we use the same meta data with different mapping function, twin peak mapping, with the corresponding 3-D data is shown in Figure 3. The four colors in both mappings represents the four gaussian kernels in the meta data. Similar to Figure 2, four manifold learnings are obtained and used to detect outliers with two bandwidth selection methods in Figure 4.

3.2 5-D data embedding in 100-D space

For the high-dimensional case, we generate the meta data from a 5-dimensional semi-hypersphere. First, we simulate $N = 2000$ points from a 4-dimensional Gaussian mixture model with two mixture components, $\mathrm{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $\mathrm{N}(\boldsymbol{\mu}_2, \sigma_2)$, where $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = (0,0,0,0)'$, $\boldsymbol{\Sigma}_1 = diag(1,1,1,1)$, and $\boldsymbol{\Sigma}_1 = diag(25,25,25,25)$. In order to manually generate anomalies, the mixture proportions are set as $\pi_1 = 0.99$ and $\pi_2 = 0.01$. The fifth dimension is calculated to satisfy $x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 = r^2$ where r is set as 15. Similarly, the Gaussian mixture densities could be calculated using Equation (1) as the true density of the 5-d meta data. Then we initial the other 95 dimensions in the high-dimensional space as zero columns and further rotate the 100-dimensional data of size N (denote the transpose of the data matrix as X) to get rid of the zeros so that it could be passed to the manifold learning algorithms. The rotation matrix is derived from the QR decomposition of a $100 \times N$ matrix A with all components randomly generated from a uniform distribution $\mathcal{U}(0,1)$. For any real matrix A of dimension $p \times q$, the QR decomposition could decompose the matrix into the multiplication

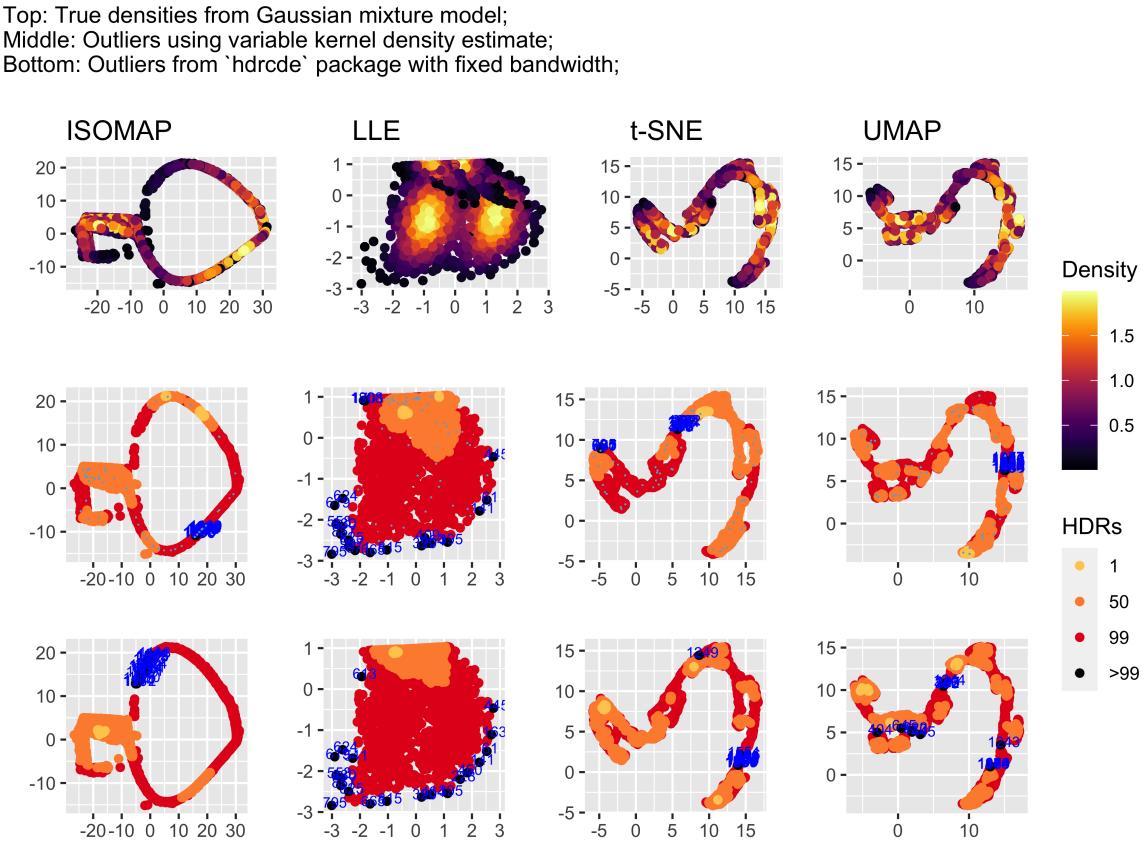

Figure 2: *Highest density region plots of four manifold learning embeddings of the swiss roll data. Colors are indicating densities from top: true densities from the Gaussian mixture model; middle: KDE with Riemannian matrix as variable bandwidth; and bottom: KDE with fixed bandwidth. Variable KDE preforms better in finding kernel structures with ISOMAP and LLE, and in locating outliers with t-SNE. Both methods are not detecting outliers accurately when using UMAP.*

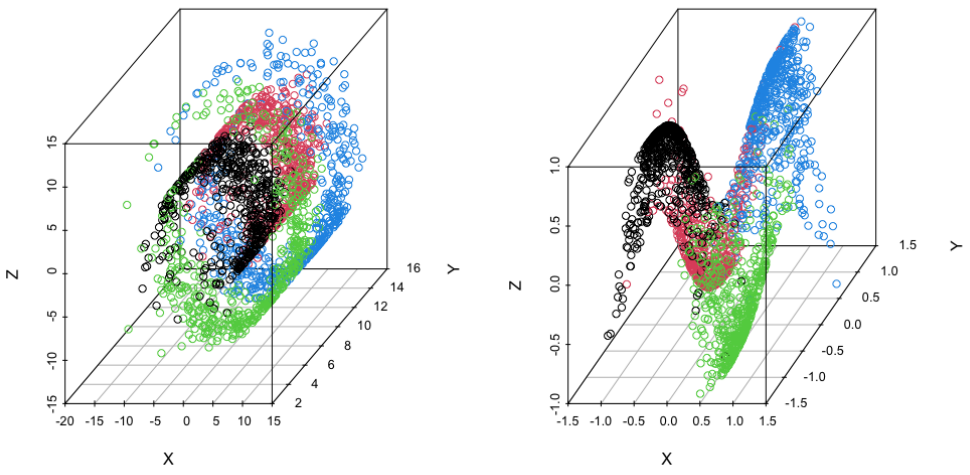

Figure 3: *3-D Mappings of the meta data with colors indicating four kernels. Left: swiss roll mapping. Right: twin peak mapping.*

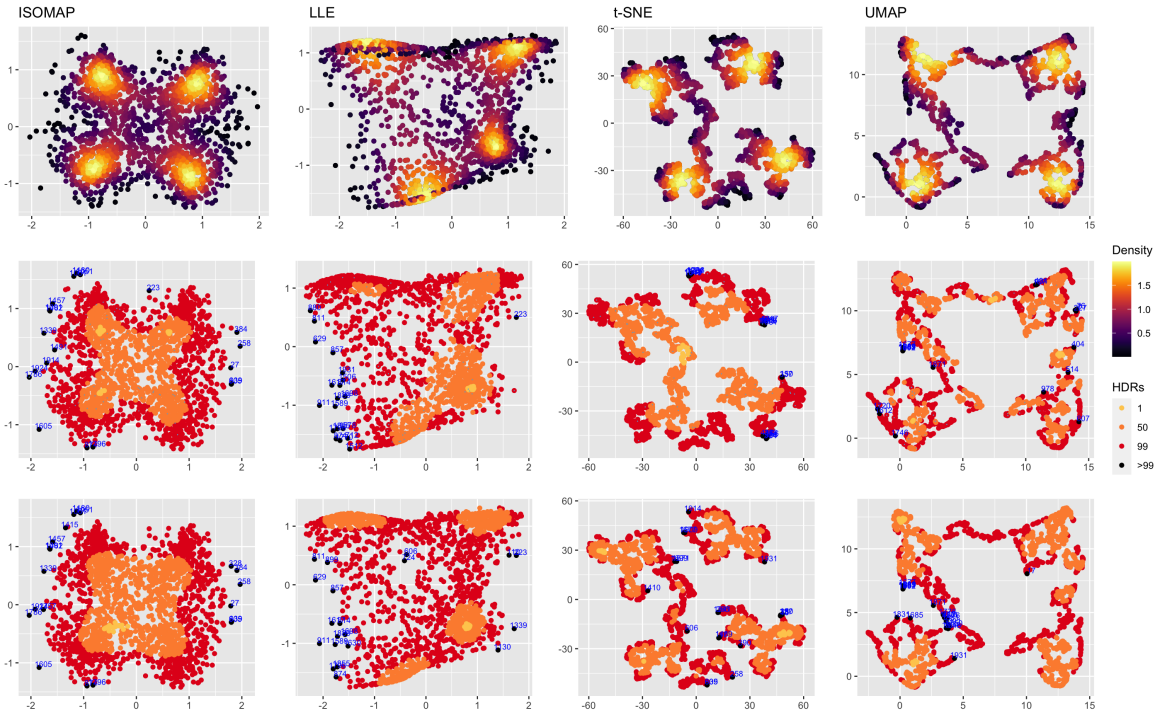

Figure 4: *Highest density region plots of four manifold learning embeddings of the twin peak data. Variable KDE preforms better in finding kernel structures with ISOMAP and LLE, and in locating outliers with t-SNE and UMAP.*

of two matrix Q and R so that $A = QR$, where the dimension of Q is a matrix with unit norm orthogonal vectors, $Q'Q = I$, and R is an upper triangular matrix. Matrix Q is used as the rotation matrix and it satisfies $X'X = (QX)'(QX)$, where the rotated data matrix $(QX)'$ is now the input for the manifold learning algorithms.

4 Application

`Dataset`

`Goal and process`

`Results (Plots and tables)`

`Analysis and summary`

5 Conclusion

`Summary of the contribution and conclusion`

Details about the process

Shortcoming and other references for future direction (improvement on density estimation,