可视化实验三报告

201900161140 张文浩

实验完成时间：10.12

软件环境：matlab

实验要求：

1， 体验 https://projector.tensorflow.org/

2， 用 https://lvdmaaten.github.io/drtoolbox/，比较 t-sne pca, isomap 等方法的区别

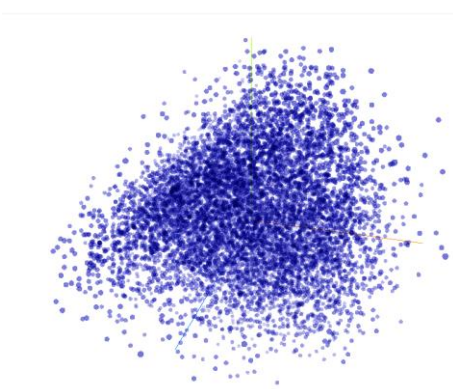
数据可以到这里下载 https://archive.ics.uci.edu/ml/datasets.php 利用 matlab 中的 drtoolbox 工具

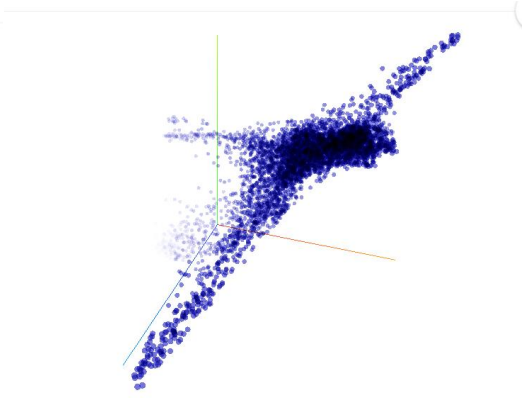
实验步骤：

1.体验 https://projector.tensorflow.org/

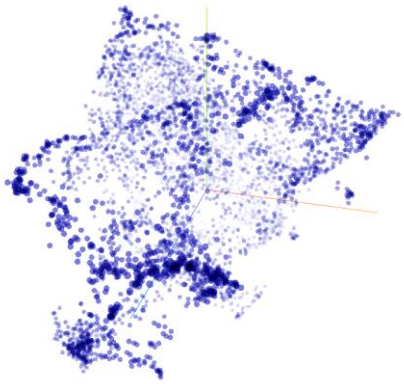
先用默认的数据集试一试

PCA： t_sne





UMAP：



在用老师给的网站的第一个数据集 abalone 数据集试一试

这个数据集是由 8 个属性，即 8 维，一共 4177 个数据

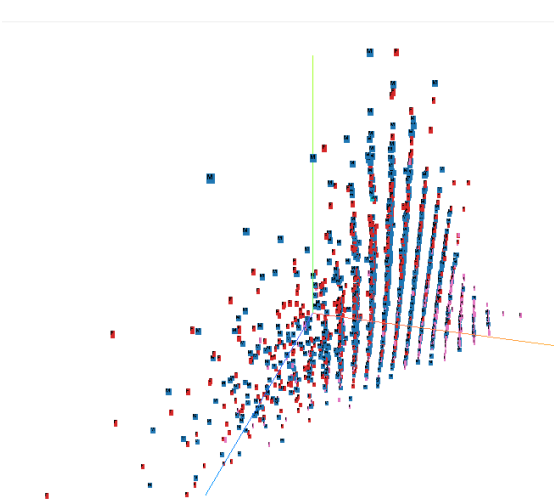
data: label:

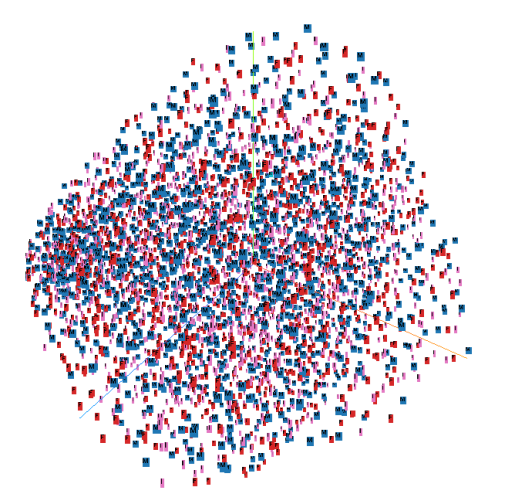
Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0.455	0.365	0.095	0.5140	0.2245	0.1010	0.1500	15
0.350	0.265	0.090	0.2255	0.0995	0.0485	0.0700	7
0.530	0.420	0.135	0.6770	0.2565	0.1415	0.2100	9
0.440	0.365	0.125	0.5160	0.2155	0.1140	0.1550	10
0.330	0.255	0.080	0.2050	0.0895	0.0395	0.0550	7
...	...	...	...	...	...	...	...
0.565	0.450	0.165	0.8870	0.3700	0.2390	0.2490	11
0.590	0.440	0.135	0.9660	0.4390	0.2145	0.2605	10
0.600	0.475	0.205	1.1760	0.5255	0.2875	0.3080	9
0.625	0.485	0.150	1.0945	0.5310	0.2610	0.2960	10
0.710	0.555	0.195	1.9485	0.9455	0.3765	0.4950	12

```
0       M
1       M
2       F
3       M
4       I
       ..
4172    F
4173    M
4174    M
4175    F
4176    M
Name: Sex, Length: 4177,
```

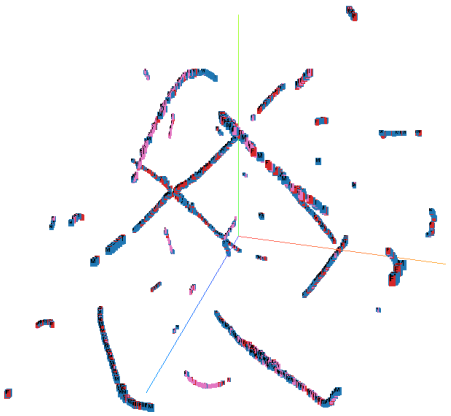
load 导入数据

PCA t_sne





UMAP



2. 用 https://lvdmaaten.github.io/drtoolbox/，比较 t-sne pca, isomap 等方法的区别：

PCA、isomap、t_sne、GDA、DiffusianMaps、LLE、laplacian、LPP、NPE、LLTSA、LLC、HessianLLE、LTSA 、ManifoldChart

PCA:

PCA（Principal Component Analysis）即主成分分析是最常见的降维方法，它是一种统计方法。用于高维数据集的探索与可视化，还可用于数据的压缩和预处理。可通过正交变换把具有相关性的高维变量转换为线性无关的低维变量，这组低维变量称为主成分，它能保留原始数据的信息。

1）输入：样本集 D={x1,x2,⋯,xn};低维空间数 d'

2）过程：

①对所有样本进行中心化;

②计算样本的协方差矩阵 XX^(T);

③对协方差矩阵 XX^(T)作特征值分解;

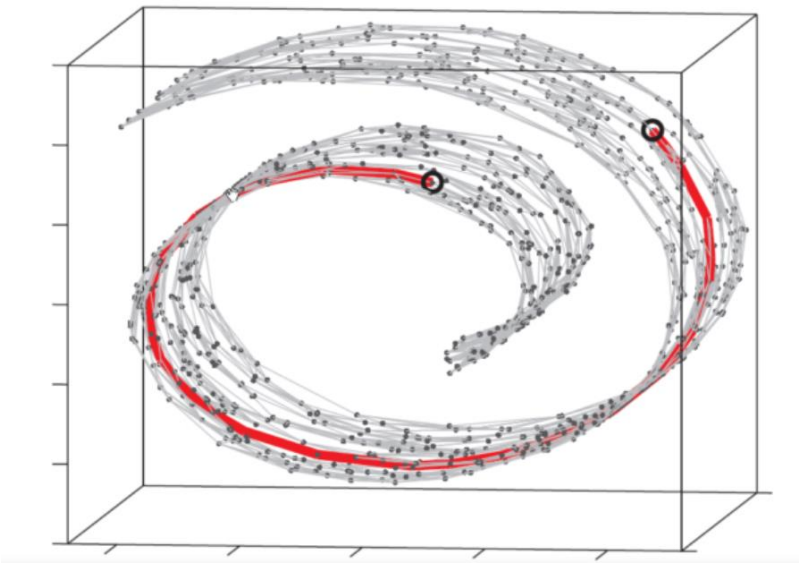
④取最大的 d'个特征值所对应的特征向量 w1,w2,⋯,w(d')

3）输出：

投影矩阵 W=（w1,w2,⋯,w(d')）

isomap：

把 MDS 中原始空间中距离的计算从欧氏距离换为了流形上的测地距离。当然，如果流形的结构事先不知道的话，这个距离是没法算的，于是 Isomap 通过将数据点连接起来构成一个邻接 Graph 来离散地近似原来的流形，而测地距离也相应地通过 Graph 上的最短路径来近似了。如下图所示：



这个东西叫做 Swiss Roll ，姑且把它看作一块卷起来的布好了。图中两个标黑圈的点，如果通过外围欧氏空间中的欧氏距离来计算的话，会是挨得很近的点，可是在流形上它们实际上是距离很远的点：红色的线是 Isomap 求出来的流形上的距离。可以想像，如果是原始的 MDS 的话，降维之后肯定会是很暴力地直接把它投影到二维空间中，完全无视流形结构，而 Isomap 则可以成功地将流形“展开”之后再做。

t_sne

设有 m 条 n 维数据:

1.将原始数据按列组成 m 行 n 列矩阵$X^T = x_1, ..., x_n$

2.计算 cost function 的参数：困惑度 perplexity

3.优化参数: 设置迭代次数 T, 学习速率, 动量

4.目标结果是低维数据表示$Y^T = y_1, ..., y_m$

5.开始优化

- 配合高维空间欧氏距离，在给定的perplexity下查找最佳σ_i使得$\text{perplexity} = H(P_i)$，条件概率$p_{j|i}$
- 令$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2}$
- 用$N(0, 10^{-4}I)$随机初始化 Y
- 从 t = 1 到 T迭代， 做如下操作:
  - 计算低维空间的$q_{ij} = \frac{(1+ \| y_i - y_j \|^2)^{-1}}{\sum_{k \neq l}(1+ \| y_i - y_j \|^2)^{-1}}$
  - 计算梯度$\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1+ \| y_i - y_j \|^2)^{-1}$
  - 更新$Y^t = Y^{t-1} + \eta \frac{dC}{dY} + \alpha(t)(Y^{t-1} - Y^{t-2})$

优化过程中可以尝试的两个 trick:

1.提前压缩(early compression):开始初始化的时候，各个点要离得近一点。这样小的距离，方便各个聚类中心的移动。可以通过引入 L2 正则项(距离的平方和)来实现。

1.提前夸大(early exaggeration)：在开始优化阶段，乘以一个大于 1 的数进行扩大，来避免因为太小导致优化太慢的问题。比如前 50 次迭代，乘以 4

剩下的降维可视化方法只简单地学习了一下概念

GDA:

域名生成算法 Domain Generation Algorithm

DiffusianMaps：

扩散映射是一种降维方法

1. 其通过 整合数据的局部几何关系 揭示 数据集在不同尺度的几何结构。

2. 与 PCA (principal component analysis)、MDS (Multidimensional Scaling) 这些降维方法相比，扩散映射 非线性，聚焦于发现数据集潜在的流形结构。

3. 优点：对噪声鲁棒，计算代价较低

LLE：局部线性嵌入

laplacian：拉普拉斯特征映射

LPP：局部保留投影

NPE：邻域保留投影算法

LLTSA：保形本征映射

LLC：本地线性协调

HessianLLE：保持邻域关系的增量

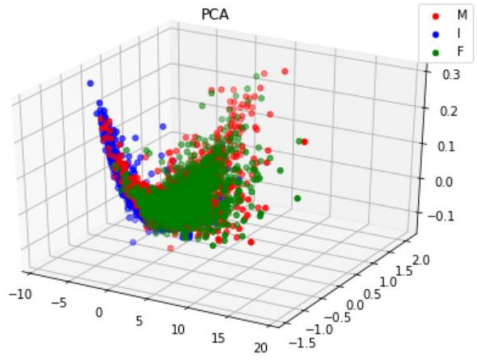
LTSA ：局部切空间排列

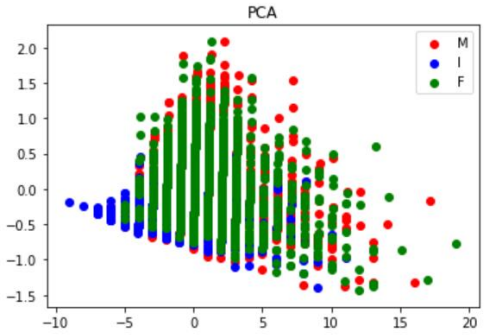
ManifoldChart：歧管图表

我在老师给的网站上下载了一个有八个维度的数据集，尝试用不同的方法降

维可视化。一开始用 matlab 实现的时候其中的 isomap 出现了问题，在降维的过程中数据集的个数被减少了，我无法把降维后的数据与 label 对应，于是改为 Python 实现。下面分别是在 python 环境中用 pca、isomap、t_sne 三种不同方法降到 3 维和 2 维的可视化效果。

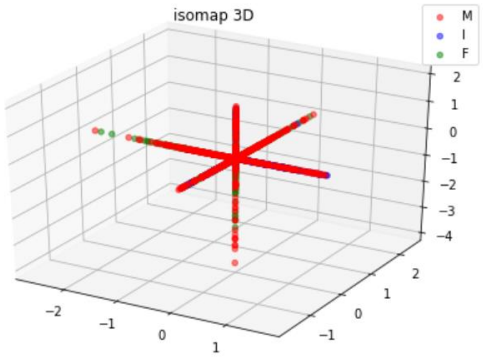
PCA：

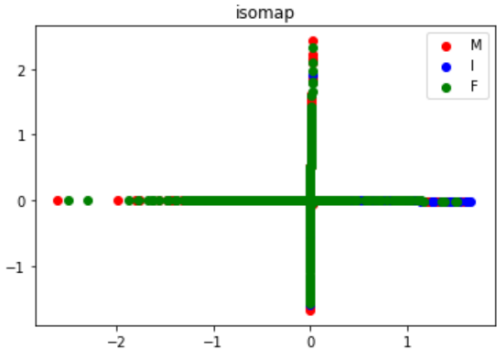




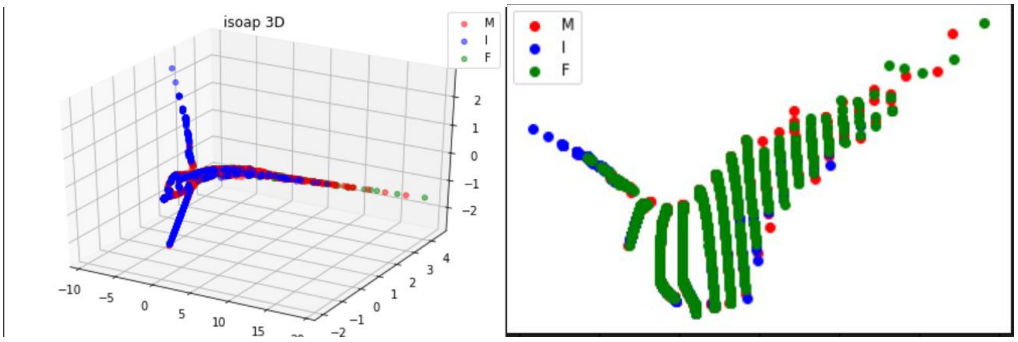
isomap：

n_neighbors = 3：

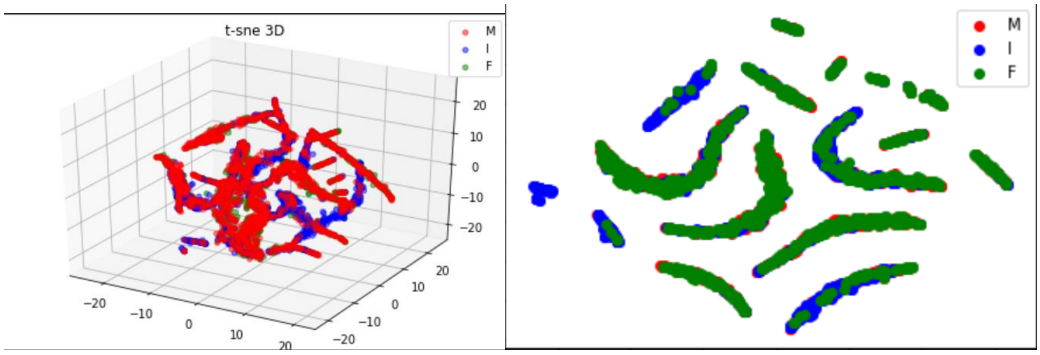




n_neighbors=300



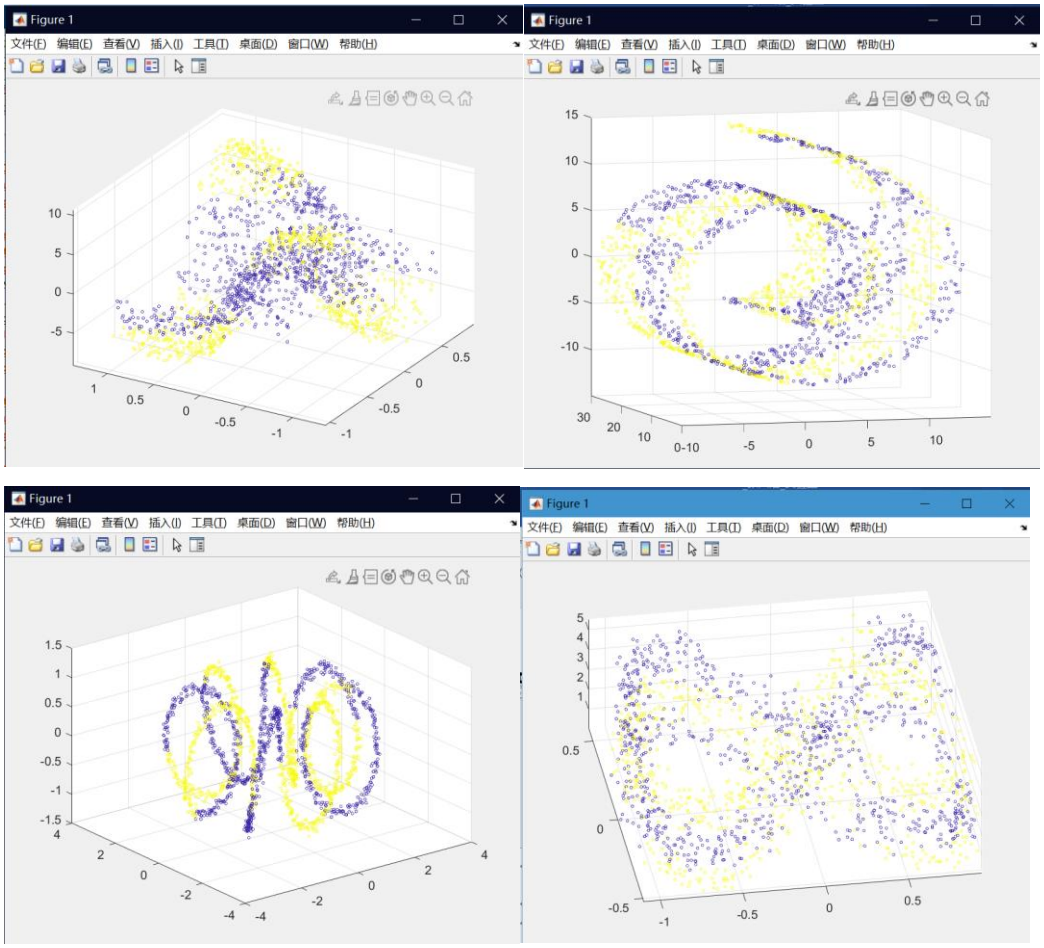
t_sne



为了尝试不同的降维可视化的方法，我使用 matlab 中 drtoolbox 自带的四个三维数据集用了 14 种降维可视化的方法降到 2 维可视化

我使用了四个数据集进行实验，分别是：twimpeaks、swiss、helix、intersect

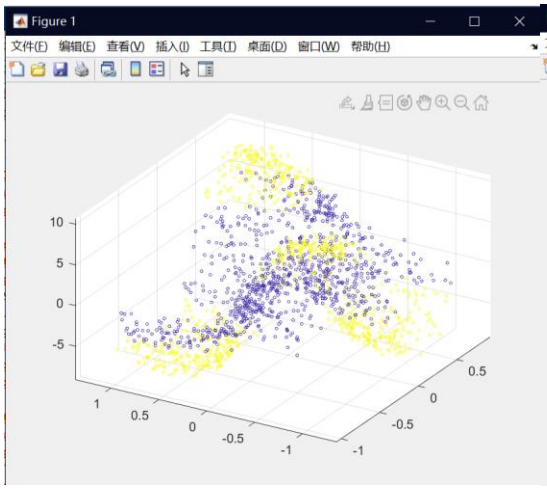
它，它们的原始数据集本来长这样：



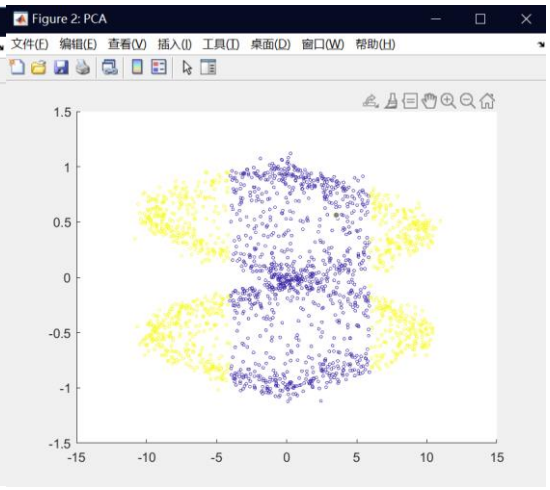
然后分别使用 14 种方式对他们进行可视化处理，比较不同的可视化方法之间的区别。14 种方法分别是：PCA、isomap、t_sne、GDA、DiffusianMaps、LLE、laplacian、LPP、NPE、LLTSA、LLC、HessianLLE、LTSA 、ManifoldChart

twimpeaks

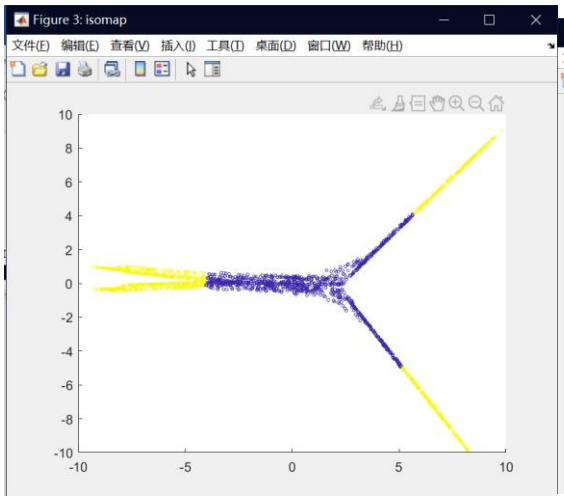
原图：



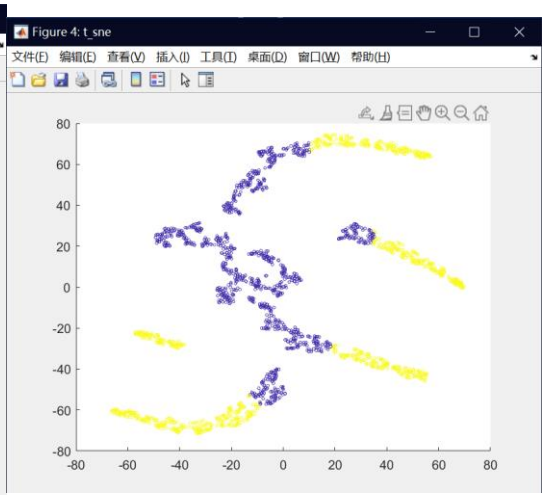
PCA：



isomap：

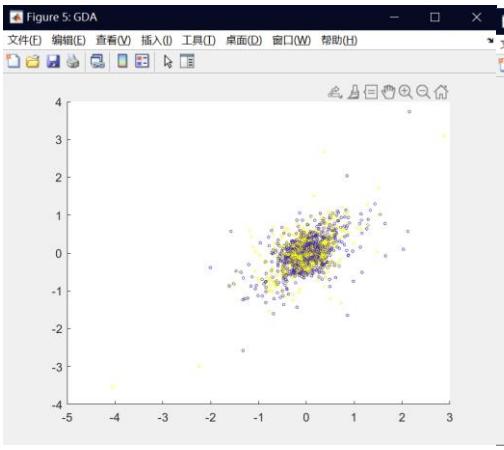


t_sne

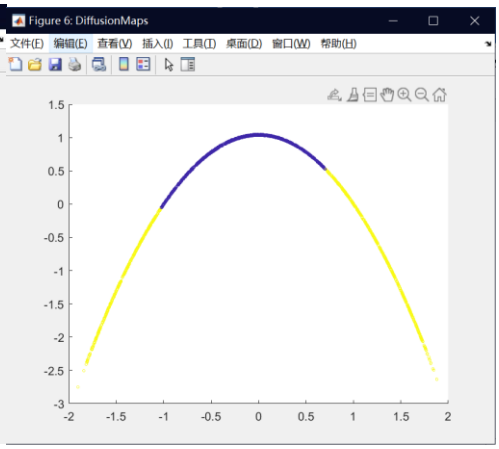


GDA：

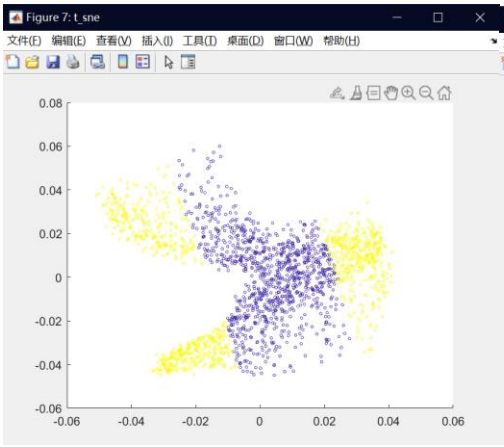
DiffusianMaps



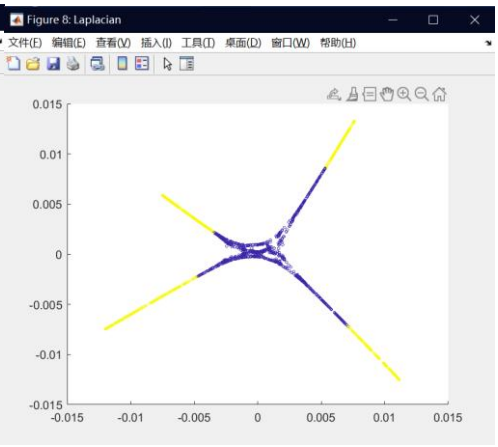
LLE:



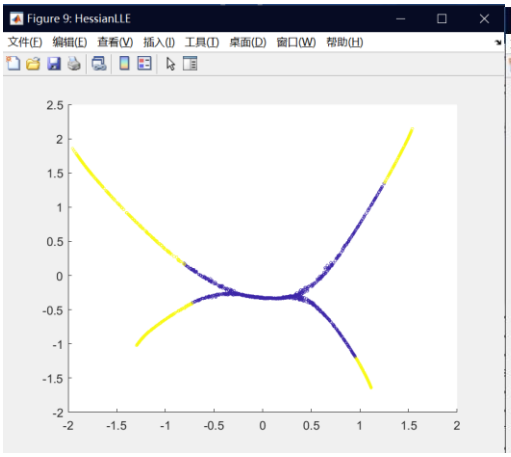
laplacian:



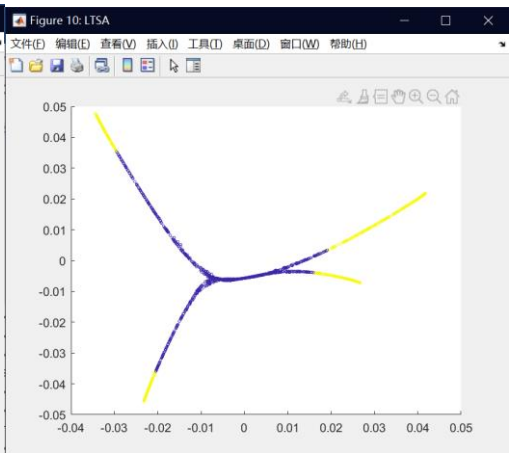
HessianLLE:



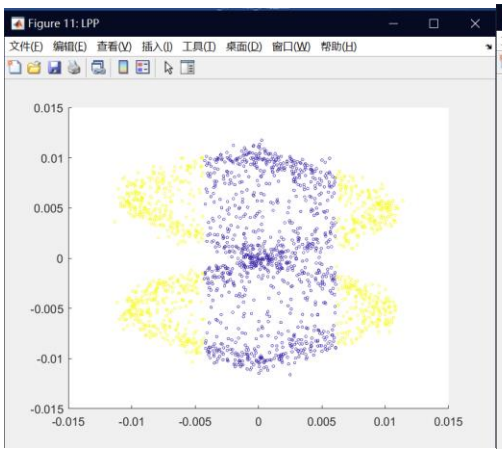
LTSA:

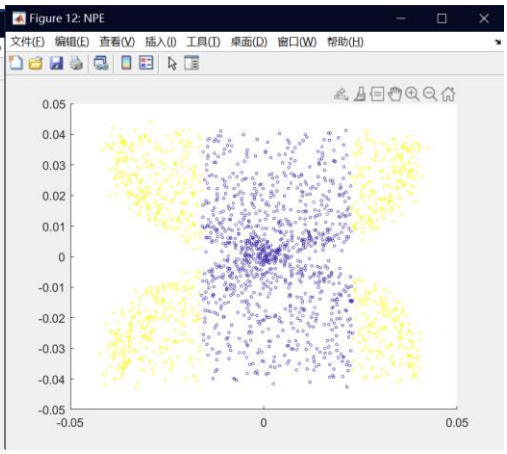


LPP:



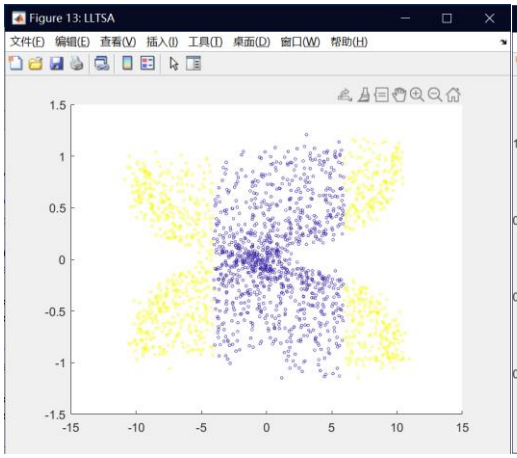
NPE:

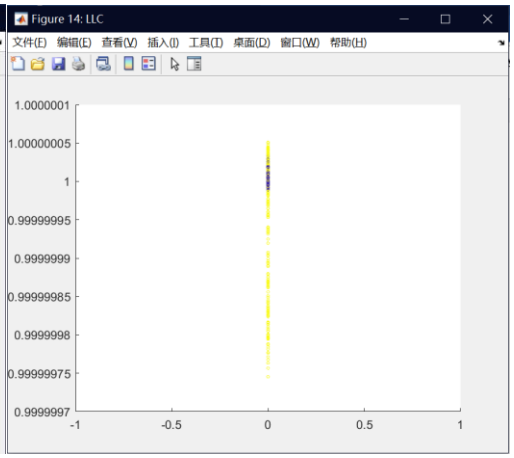




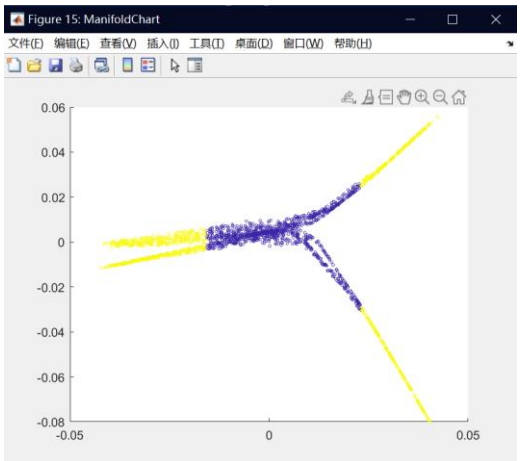
LLTSA:

LLC:





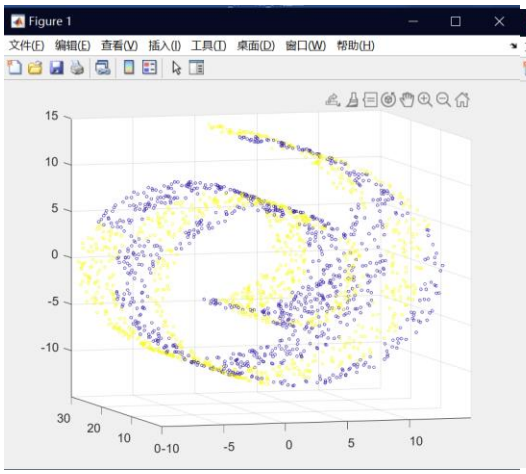
ManifoldChart:



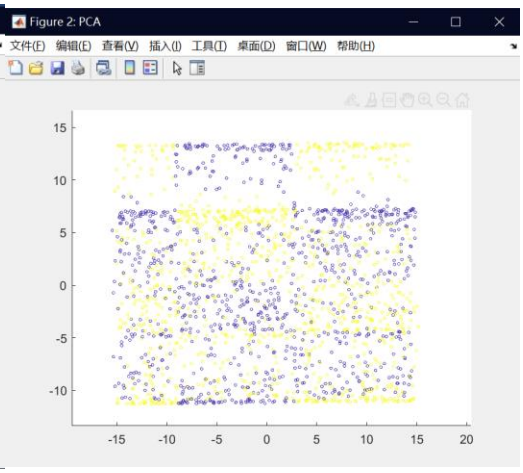
swiss:

原图:

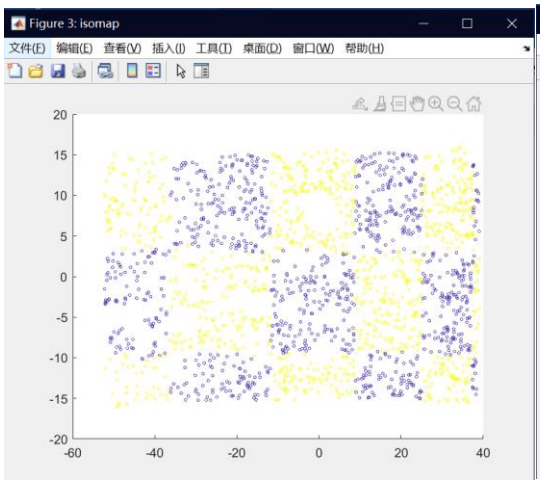
PCA:



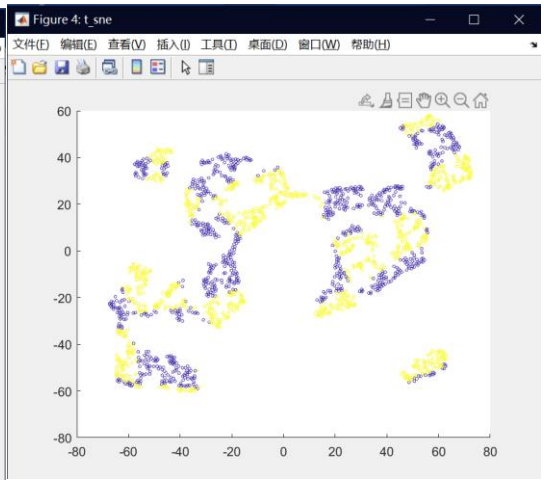
isomap:



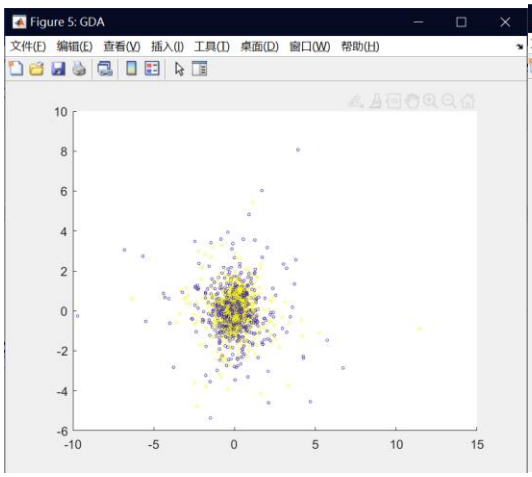
t_sne:



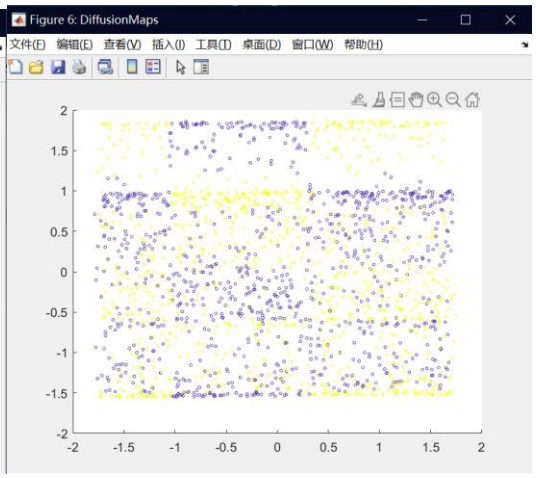
GDA:



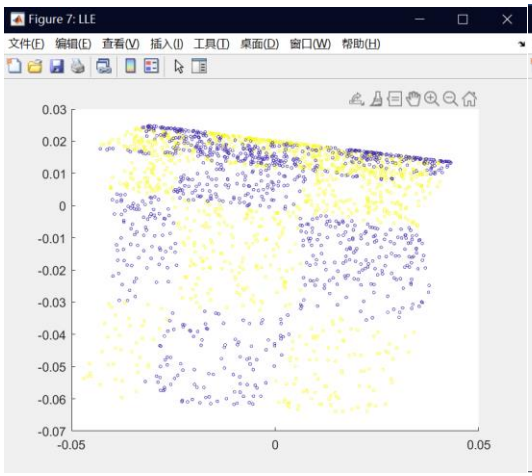
DiffusionMaps:

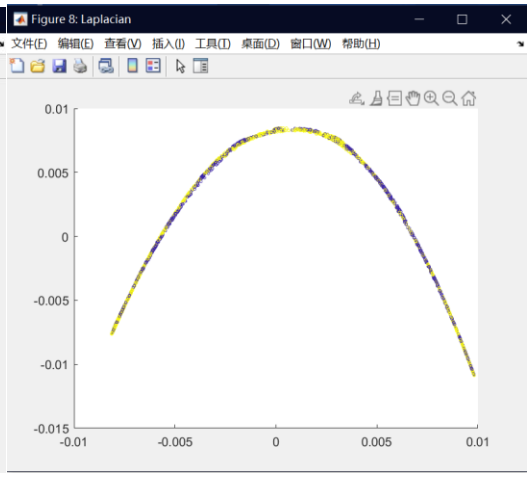


LLE:



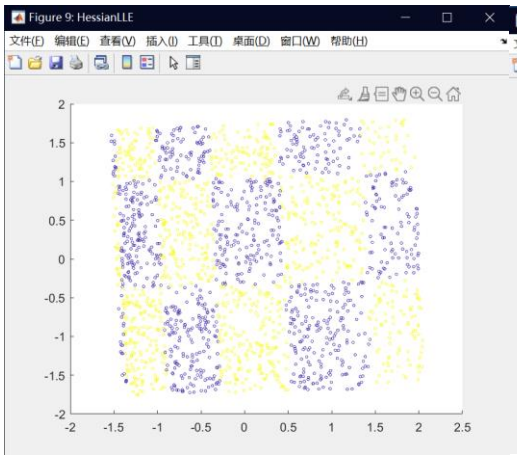
Lapiacian:

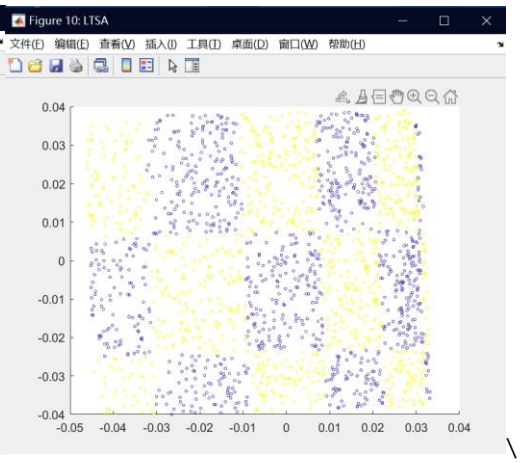




HessianLLE:

LTSA:

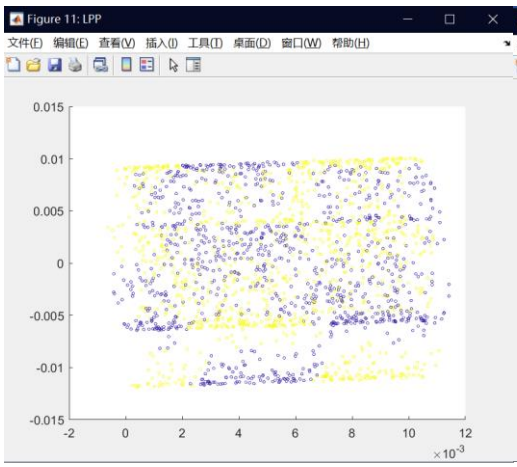


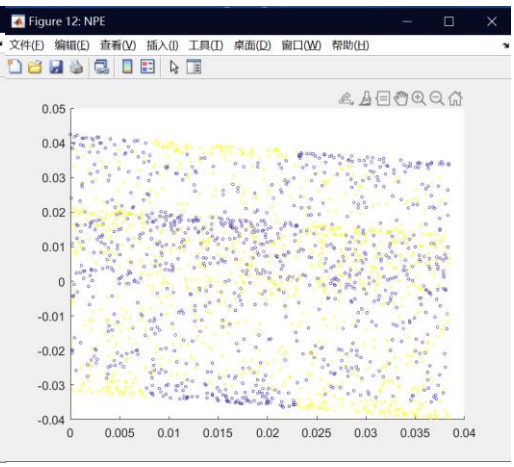


\

LPP:

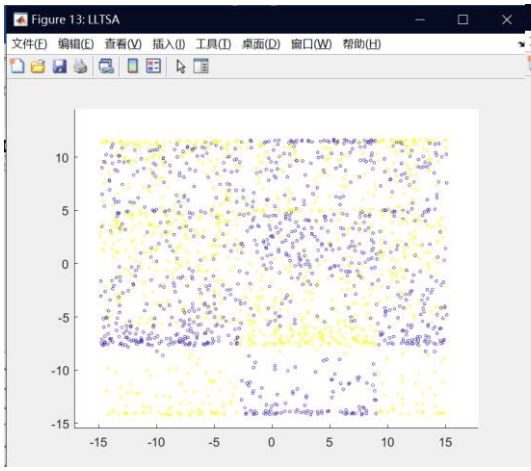
NPE:

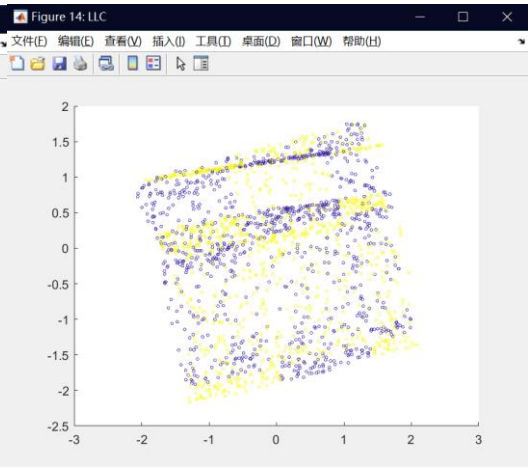




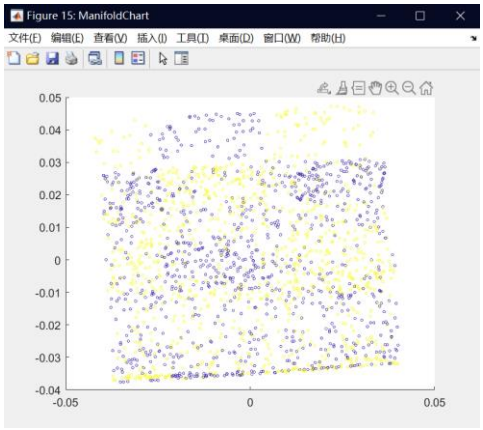
LLTSA:

LLC:



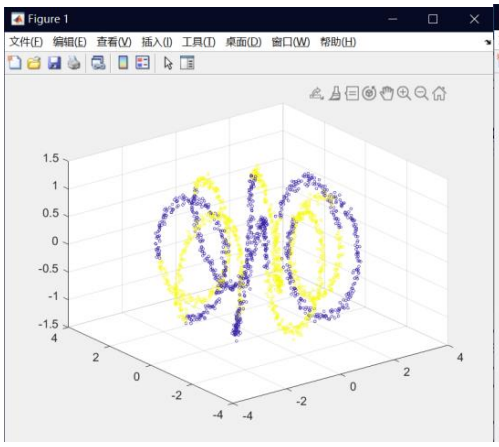


ManifoldChart

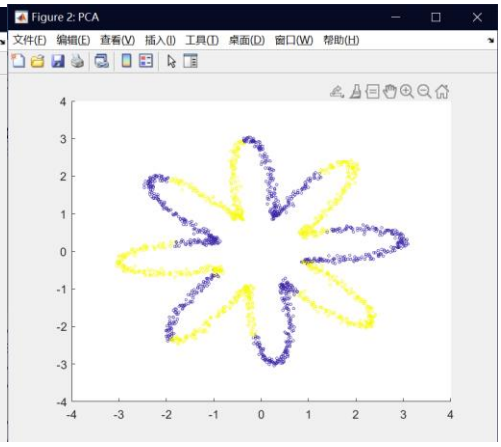


helix：

原图：

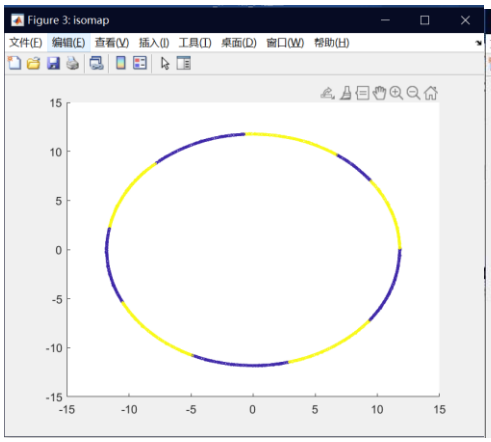


PCA：

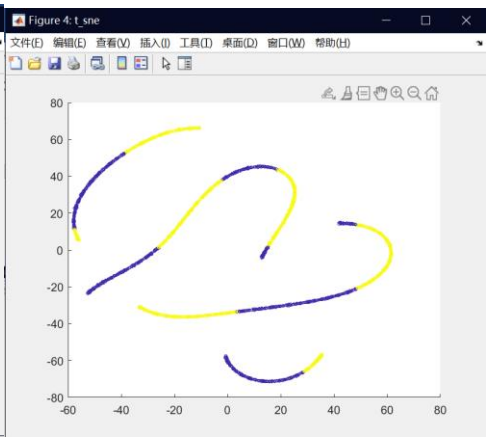


isomap：

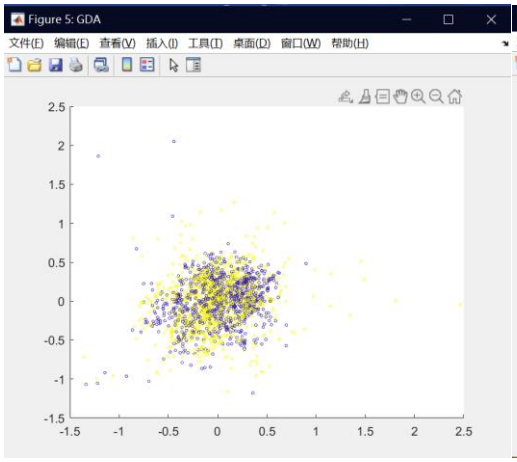
t_sne



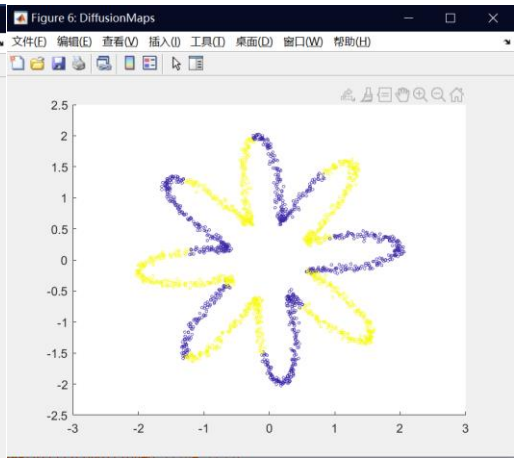
GDA:



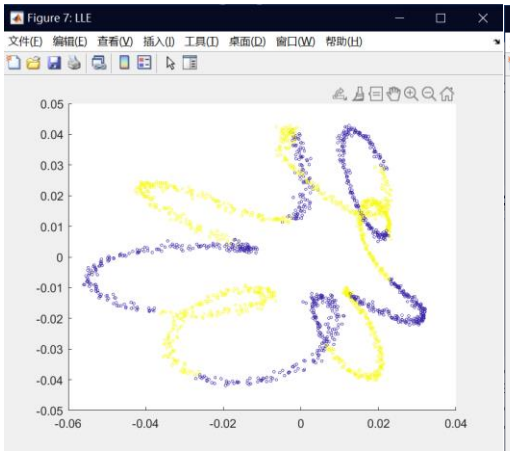
DiffusionMaps:



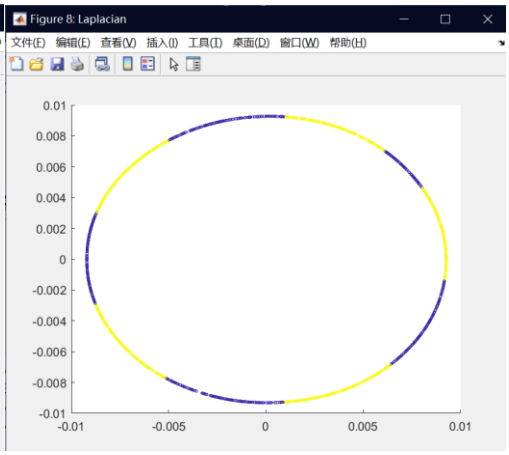
LLE:



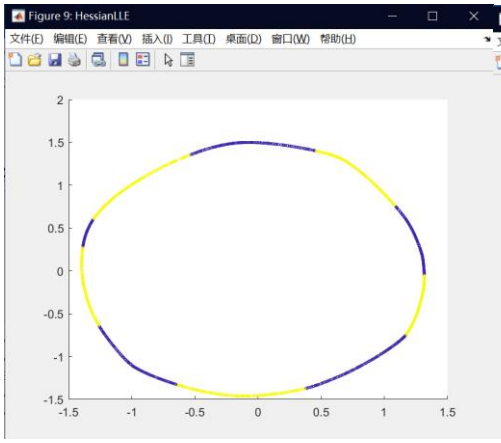
Laplacian:

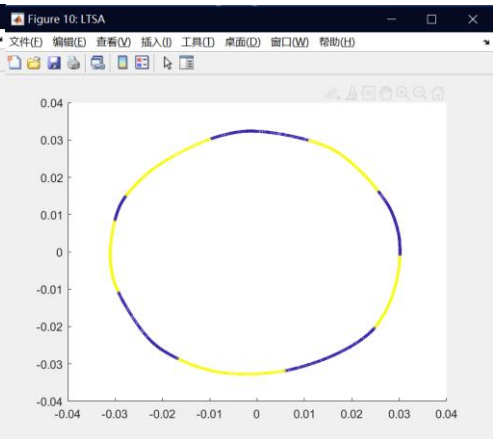


HessianLLE:



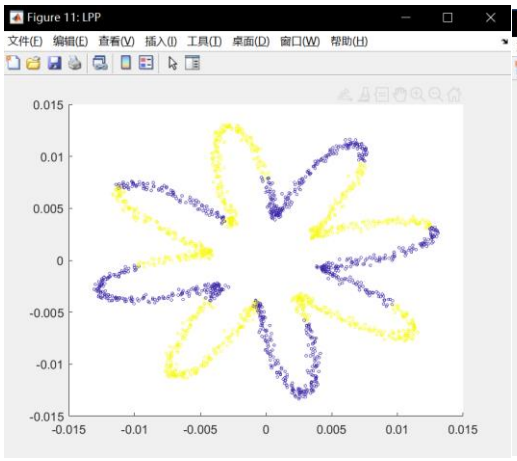
LTSA

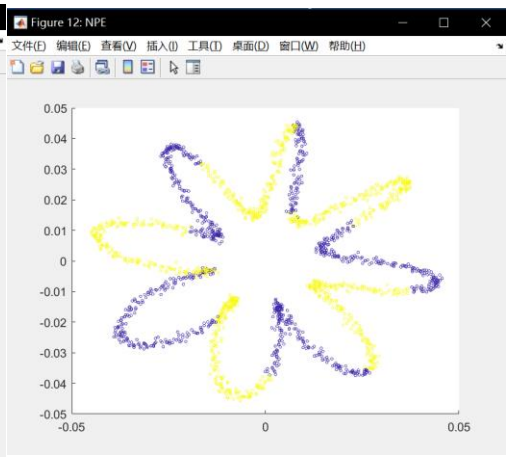




LPP:

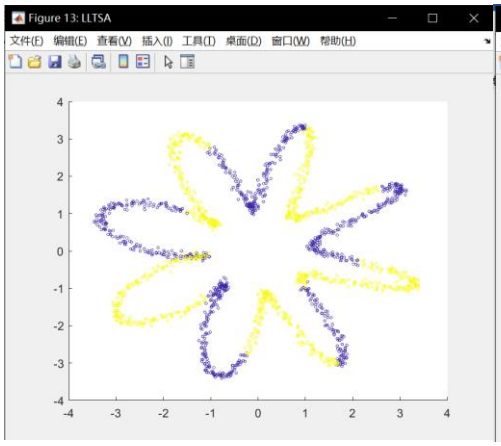
NPE:

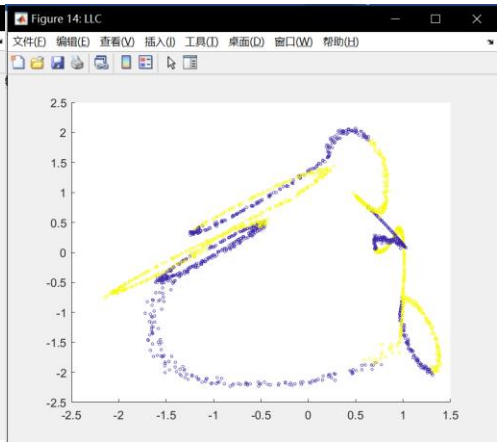




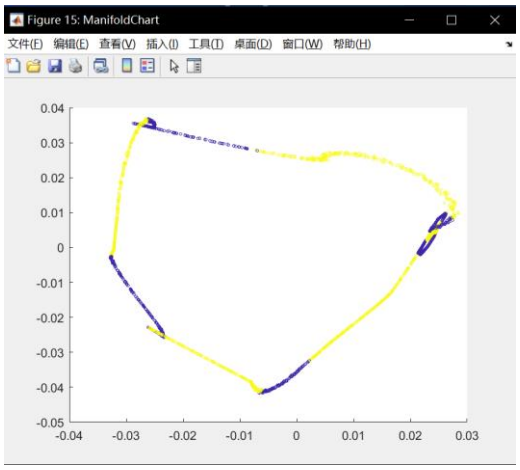
LLTSA:

LLC:



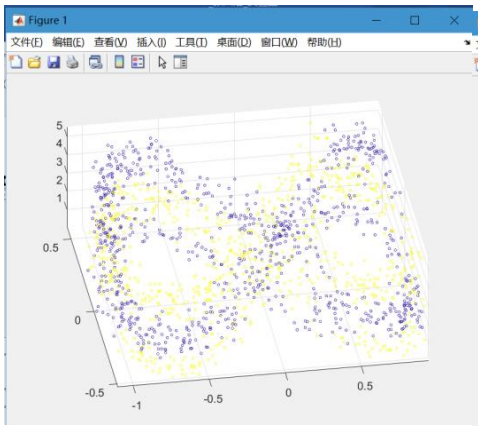


ManifoldChart:

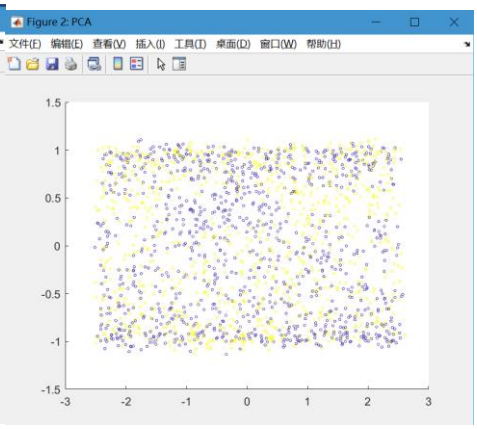


intersect:

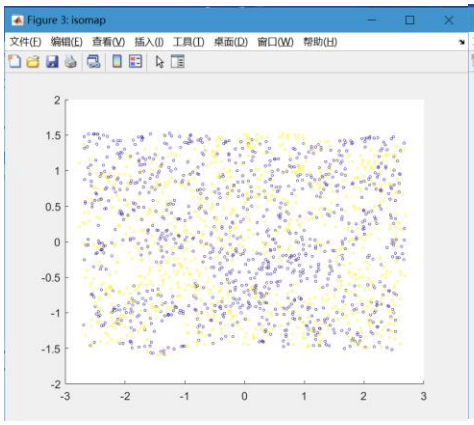
原图：



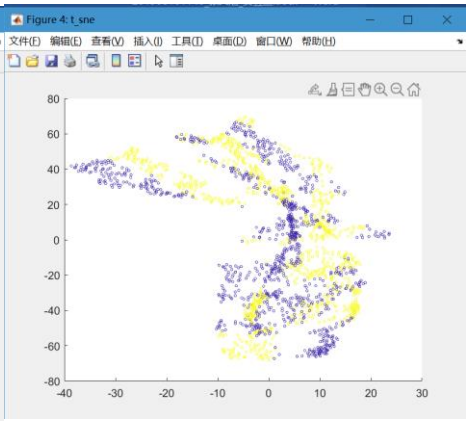
PCA：



isomap：

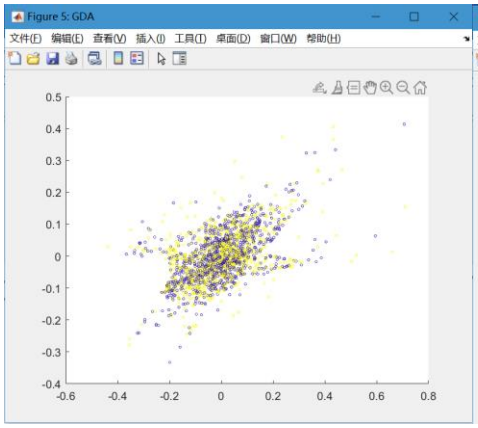


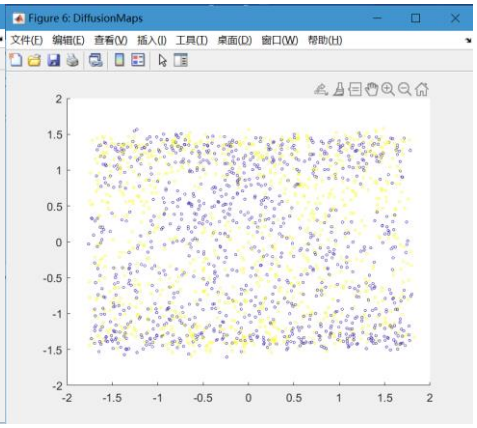
t_sne:



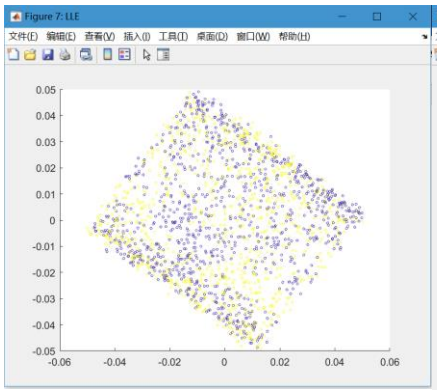
GDA:

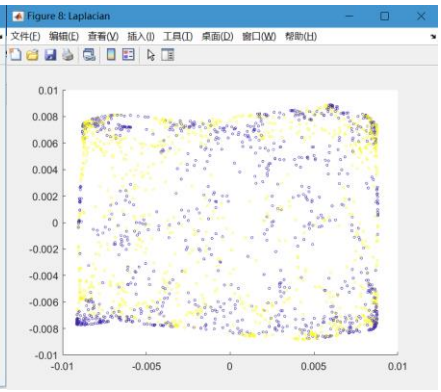
DiffusionMaps:



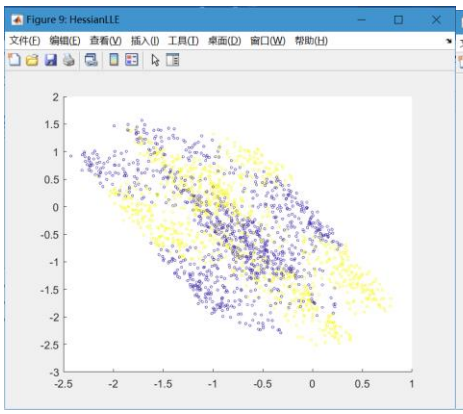


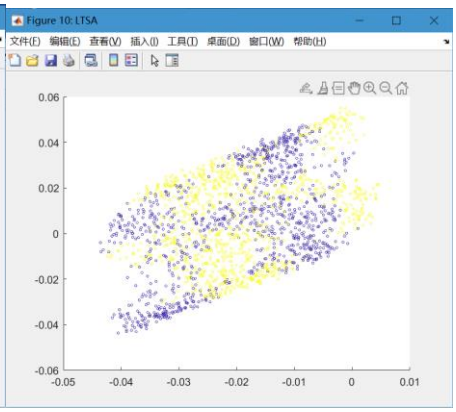
LLE: Laplacian:



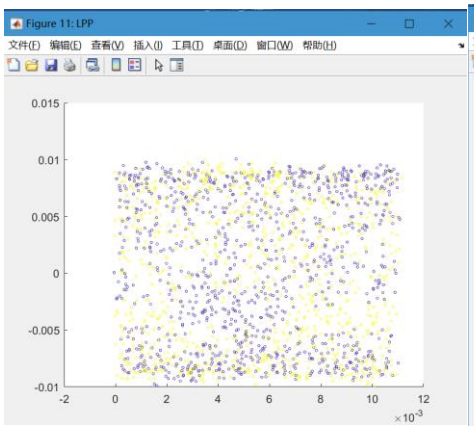


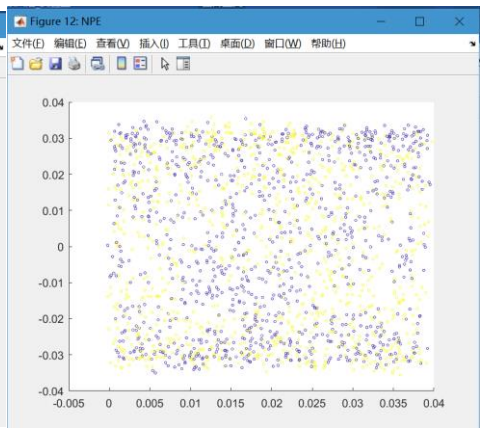
HessianLLE: LTSA:



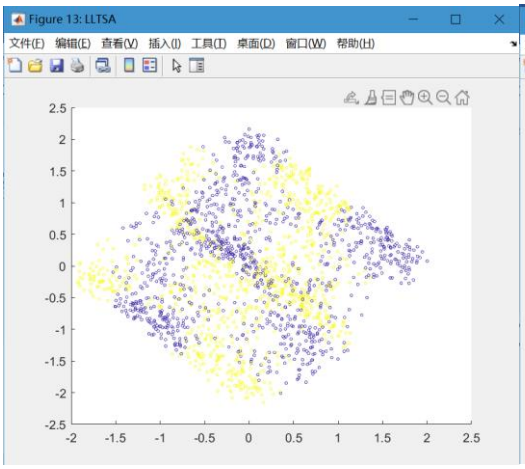


LLP: NPE:

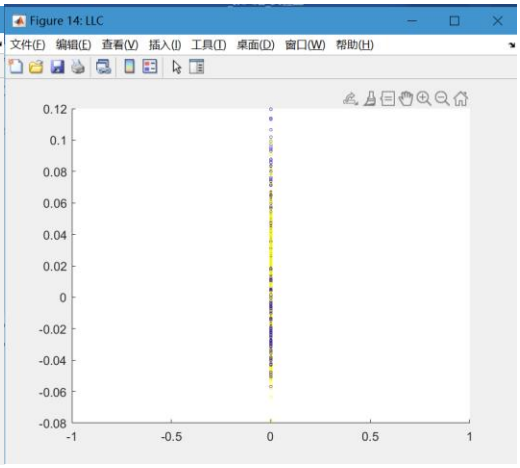




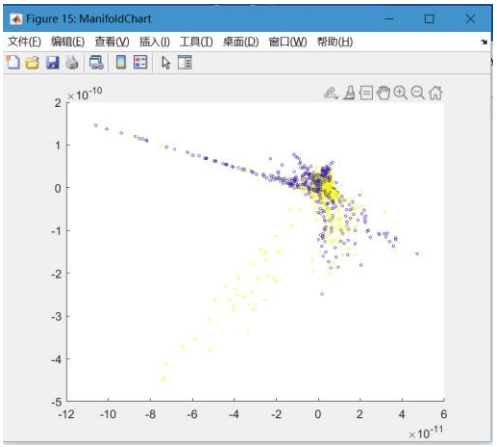
LLTSA:



LLC:



ManifoldChart



实验总结

PCA 是大家所熟知的降维算法，但是线性降维虽然简单，其局限性也很明显，难以实现高维数据在低维空间的可视化。

t-SNE 是非线性的降维算法，能实现高维到低维的可视化映射，但因为涉及大量的条件概率、梯度下降等计算，时间和空间复杂度是平方级的，比较耗资源。

isomap 对于流形（Manifold，局部具有欧式空间性质的空间），两点之间的距离并非欧氏距离。而是采用“局部具有欧式空间性质”的原因，让两点之间的距离近似等于依次多个临近点的连线的长度之和。通过这个方式，将多维空间“展开”到低维空间。