降维算法

陈 鑫

2023 年 4 月 29 日

目录

目录

降维算法基本概念

降维算法是聚类算法之外的另外一类重要的无监督学习算法，其主要任务是对高维的特征做低维近似。高维特征不仅耗费大量的存储空间，还会降低机器学习的时间和空间效率，通过降维，提升算法的效率，也便于数据的可视化，进而更好地理解数据。在降维过程中，如何尽可能保留原特征所携带的重要信息，是降维算法主要考虑的问题。

常见的降维算法包括有主成分分析法，主成分分析的核方法，线性判别分析法，流形降维算法，自动编码器等。

在一个降维问题中，训练数据为 m 个 n 维向量 $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(m)} \in \mathbb{R}^n$。对指定的维度 $d < n$，一个降维模型是从 $\mathbb{R}^n$ 到 $\mathbb{R}^d$ 的映射 h，其模型输出为 $h(\boldsymbol{x}^{(1)}), h(\boldsymbol{x}^{(2)}), \cdots, h(\boldsymbol{x}^{(m)}) \in \mathbb{R}^d$。对模型的不同度量方式会导出不同的降维算法。主成分分析法（Principle Component Analysis, PCA）就是一种降维算法，它以降维后数据的方差作为模型的度量。主成分分析法将数据投影至低维空间，并使其投影方差尽可能大，因为方差能表示一组数据的信息量，因此，主成分分析法能最大程度地保留原始数据的信息量。一个简单的例子说明这一点。simple_example_projection.py

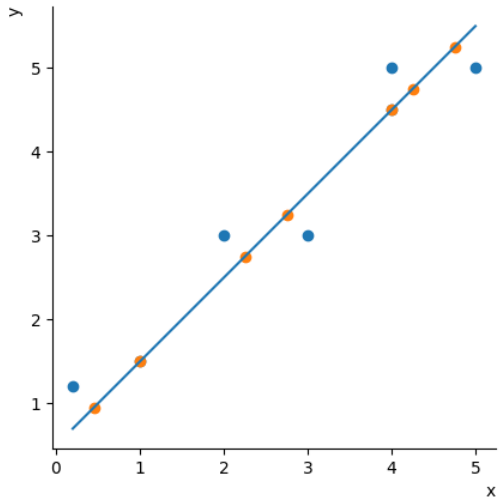


图 1: 将二维空间的数据投影至一维空间

给定一组 n 维向量 $\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)},\cdots,\boldsymbol{x}^{(m)} \in \mathbb{R}^n$ 以及一个单位向量 $\boldsymbol{w} \in \mathbb{R}^n$。定义 $z_t = \boldsymbol{x}^{(t)T}\boldsymbol{w}$ 为 $\boldsymbol{x}^{(t)}$ 沿着 $\boldsymbol{w}$ 方向的投影，并将 $z_1, z_2, \cdots, z_m$ 的方差

$$\sigma^2 = \frac{1}{m}\sum_{t=1}^{m}(z_t - \mu)^2$$

定义为 $\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)},\cdots,\boldsymbol{x}^{(m)}$ 沿着 $\boldsymbol{w}$ 方向的投影方差，其中 $\mu = \frac{1}{m}\sum_{t=1}^{m} z_t$ 为 $z_1, z_2, \cdots, z_m$ 的平均值。

在一个降维问题中，主成分分析法按投影方差最大的原则将 m 个 n 维向量组成的训练数据 $\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)},\cdots,\boldsymbol{x}^{(m)}$ 投影至指定的 d 维 $(d < n)$ 空间。

为了简化计算，将向量 $\boldsymbol{x}$ 平移到均值为 0 的地方，$\boldsymbol{x} \leftarrow \boldsymbol{x} - \mu$，其中 $\mu = \frac{1}{m}\sum_{i=1}^{m}\boldsymbol{x}^{(i)}$。算法首先计算 $\boldsymbol{w}^{(1)}$，其应当是使得 $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(m)}$ 的投影方差最大的单位向量，所以 $\boldsymbol{w}^{(1)}$ 是以下带约束的最优化问题的最优解:

$$\max \frac{1}{m}\sum_{t=1}^{m}(\boldsymbol{x}^{(t)T}\boldsymbol{w}(-\mu = 0))^2$$

$$\text{满足 } ||\boldsymbol{w}||^2 = 1$$

将上式表示成为矩阵形式，并通过拉格朗日函数以及 KKT 定理，最终得到其最优值应当等于 X^TX 的最大特征根的值，$X^TX\boldsymbol{w} = \lambda\boldsymbol{w}$。因此，主成分分析法应当以如下方式选取 $\boldsymbol{w}^{(1)}$：计算 X^TX 的特征根与特征向量，设最大特征根为 λ_1，将 λ_1 对应的单位特征向量取为 $\boldsymbol{w}^{(1)}$，将 $X\boldsymbol{w}^{(1)}$ 称为 X 的第一主成分。计算出第一主成分之后，接着计算第二、第三主成分，最终得到取 $\boldsymbol{w}^{(i)}$ 为 X^TX 的第 i 大特征根 λ_i 对应的特征向量，$X^T\boldsymbol{w}^{(i)}$ 为 X 的第 i 个主成分。

主成分分析算法描述

Algorithm 1: 主成分分析算法

$\mu = \frac{1}{m}\sum_{t=1}^{m} \boldsymbol{x}^{(t)}$

for $t = 1, 2, \cdots, m$ **do**

$\quad \boldsymbol{x}^{(t)} \leftarrow \boldsymbol{x}^{(t)} - \mu$

$$X = \begin{pmatrix} \boldsymbol{x}^{(1)T} \\ \boldsymbol{x}^{(2)T} \\ \\ \vdots \\ \boldsymbol{x}^{(m)T} \end{pmatrix}$$

计算 $X^T X$ 的特征值 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, 令 $\boldsymbol{w}^{(1)}, \boldsymbol{w}^{(2)}, \cdots, \boldsymbol{w}^{(n)}$ 是特征值对应的特征向量。

$W = (\boldsymbol{w}^{(1)}, \boldsymbol{w}^{(2)}, \cdots, \boldsymbol{w}^{(d)})$

return $Z = XW$

为了评价主成分分析法的降维效果，可将降维后的数据再投影回原来的高维空间，这种投影称为数据重构。通过数据重构可以度量降维的效果。重构后的数据越接近原始数据，降维效果越好。设 $\boldsymbol{w}^{(1)}, \boldsymbol{w}^{(2)}, \cdots, \boldsymbol{w}^{(d)} \in \mathbb{R}^n$ 为 d 个相互正交的 n 维单位向量。它们张成一个 $\mathbb{R}^n$ 的 d 维线性子空间

$$S = \{z_1\boldsymbol{w}^{(1)} + z_2\boldsymbol{w}^{(2)} + \cdots + z_d\boldsymbol{w}^{(d)} | z_1, z_2, \cdots, z_d \in \mathbb{R}\}$$

因此，S 中的每一个 n 维向量都可以用一个 d 维向量 $\boldsymbol{z} = (z_1, z_2, \cdots, z_d)$ 来唯一表示。

主成分分析法选出的 $\boldsymbol{w}^{(1)}, \boldsymbol{w}^{(2)}, \cdots, \boldsymbol{w}^{(d)} \in \mathbb{R}^n$ 是降维后的 d 维空间的一组标准正交基，算法将每个 n 维向量 $\boldsymbol{x}$ 投影成 d 维向量 $\boldsymbol{z} = (z_1, z_2, \cdots, z_d)$，其中，$z_r = \boldsymbol{x}^T\boldsymbol{w}^{(r)}, r = 1, 2, \cdots, d$。$\boldsymbol{z}$ 对应于 n 维向量 $\tilde{\boldsymbol{x}} = \sum_{r=1}^{d} z_r\boldsymbol{w}^{(r)}$，也就是说，可以将 $\tilde{\boldsymbol{x}}$ 看成是 $\boldsymbol{x}$ 降维后的向量 $\boldsymbol{z}$ 在原空间的重构。

主成分分析的数据重构算法可描述如下。
设主成分分析将数据 $\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)},\cdots,\boldsymbol{x}^{(m)} \in \mathbb{R}^n$ 降维成 $\boldsymbol{z}^{(1)},\boldsymbol{z}^{(2)},\cdots,\boldsymbol{z}^{(m)} \in \mathbb{R}^d$。
定义 $\tilde{\boldsymbol{x}}^{(t)} = W\boldsymbol{z}^{(t)}, 1 \leq t \leq m$，$W = (\boldsymbol{w}^{(1)},\boldsymbol{w}^{(2)},\cdots,\boldsymbol{w}^{(d)})$。
则 $\tilde{\boldsymbol{x}}^{(1)},\tilde{\boldsymbol{x}}^{(2)},\cdots,\tilde{\boldsymbol{x}}^{(m)}$ 是对 $\boldsymbol{x}^{(1)},\boldsymbol{x}^{(2)},\cdots,\boldsymbol{x}^{(m)}$ 的重构。

$$\text{定义 } \tilde{X} = \begin{pmatrix} \tilde{\boldsymbol{x}}^{(1)T} \\ \tilde{\boldsymbol{x}}^{(2)T} \\ \vdots \\ \tilde{\boldsymbol{x}}^{(m)T} \end{pmatrix} = \begin{pmatrix} (W\boldsymbol{z}^{(1)})^T \\ (W\boldsymbol{z}^{(2)})^T \\ \vdots \\ (W\boldsymbol{z}^{(m)})^T \end{pmatrix} = \begin{pmatrix} \boldsymbol{z}^{(1)T} \\ \boldsymbol{z}^{(2)T} \\ \vdots \\ \boldsymbol{z}^{(m)T} \end{pmatrix} W^T \text{，则 } \tilde{X} = ZW^T$$

主成分分析的算法实现参考 pca.py
案例实战 10.1 数字数据集的降维。参考代码 digists_pca.py

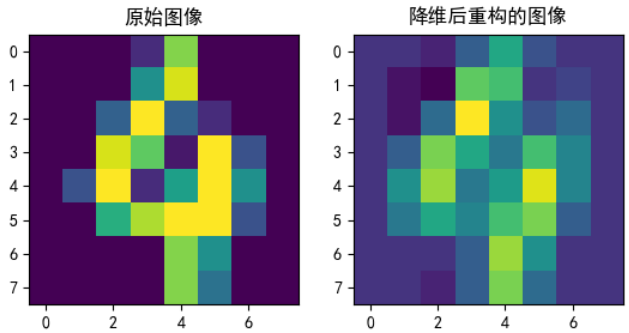


图 2: 数字数据集的降维

目录

主成分分析法是用线性投影的方式实现数据降维，它适用于高维空间中的数据近似地分布于某个低维线性子空间的情形。当数据样本的 n 个特征之间不存在线性相关性时，主成分分析法的降维效果就不是很理想，这时就需要用到主成分分析的核方法来对数据进行降维，以达到更好的效果。核方法在支持向量机中已经做了介绍，主成分分析是核方法的另一个重要应用场景。

设 $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(m)} \in \mathbb{R}^n$ 是 m 个待降维的 n 维数据，核方法通过定义一个映射 $\phi : \mathbb{R}^n \to \mathbb{R}^N$ 将一个 n 维向量映射为一个 N 维向量，使得 $\phi(\boldsymbol{x}^{(1)}), \phi(\boldsymbol{x}^{(2)}), \cdots, \phi(\boldsymbol{x}^{(m)})$ 近似地分布在 $\mathbb{R}^N$ 的某个低维线性子空间中，从而可以对 $\phi(\boldsymbol{x}^{(1)}), \phi(\boldsymbol{x}^{(2)}), \cdots, \phi(\boldsymbol{x}^{(m)})$ 进行主成分分析降维。核方法的关键是无需了解映射 ϕ 的具体形式，只要核函数具有高效的计算方法就可以了。

设降维的目标是将 n 维数据降至 d 维，取定一个映射 $\phi : \mathbb{R}^n \to \mathbb{R}^N$ 及其核函数 K_ϕ。在 $\mathbb{R}^N$ 中用主成分分析法对 $\phi(\boldsymbol{x}^{(1)}), \phi(\boldsymbol{x}^{(2)}), \cdots, \phi(\boldsymbol{x}^{(m)})$ 降维时，不妨设 $\frac{1}{m}\sum_{t=1}^{m}\phi(\boldsymbol{x}^{(t)}) = 0$，定义

$$\Phi(X) = \begin{pmatrix} \phi(\boldsymbol{x}^{(1)})^T \\ \phi(\boldsymbol{x}^{(2)})^T \\ \vdots \\ \phi(\boldsymbol{x}^{(m)})^T \end{pmatrix}$$

以及 $K = \phi(X)\phi(X)^T$，可以通过核函数计算出矩阵 K 中的元素。K 中第 t 行第 s 列的元素为 $K_{t,s} = \phi(\boldsymbol{x}^{(t)})^T\phi(\boldsymbol{x}^{(s)}) = K_\phi(\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(s)}), t, s = 1, 2, \cdots, m$。

若 $\frac{1}{m}\sum_{t=1}^{m}\phi(\boldsymbol{x}^t) \neq 0$，则定义 $m \times m$ 矩阵 $\hat{\boldsymbol{K}} = (\hat{K}_{t,s})_{1\leq t,s,\leq m}$，其 t 行 s 列的元素为

$$\begin{aligned}\hat{K_{t,s}} &= \left(\phi(\boldsymbol{x}^{(t)}) - \frac{1}{m}\sum_{i=1}^{m}\phi(\boldsymbol{x}^{(i)})\right)^T\left(\phi(\boldsymbol{x}^{(t)}) - \frac{1}{m}\sum_{j=1}^{m}\phi(\boldsymbol{x}^{(j)})\right)\\ &= K_{t,s} - \frac{1}{m}\sum_{i=1}^{m}K_{t,i} - \frac{1}{m}\sum_{j=1}^{m}K_{j,s} + \frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}K_{i,j}\end{aligned}$$

因此有 $\hat{\boldsymbol{K}} = K - JK - KJ + JKJ$，其中 J 是所有元素均为 1 的 $m \times m$ 矩阵。

为了将核函数与主成分分析法有机结合，需要采用矩阵奇异值分解对 PCA 的基本形式做等价变换，最终得到如下算法。

主成分分析的核方法

Algorithm 2: 主成分分析的核方法

for $t, s = 1, 2, \cdots, m$ **do**

$K_{t,s} = K_\phi(\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(s)})$

J 是一个 $m \times m$ 的全 1 矩阵

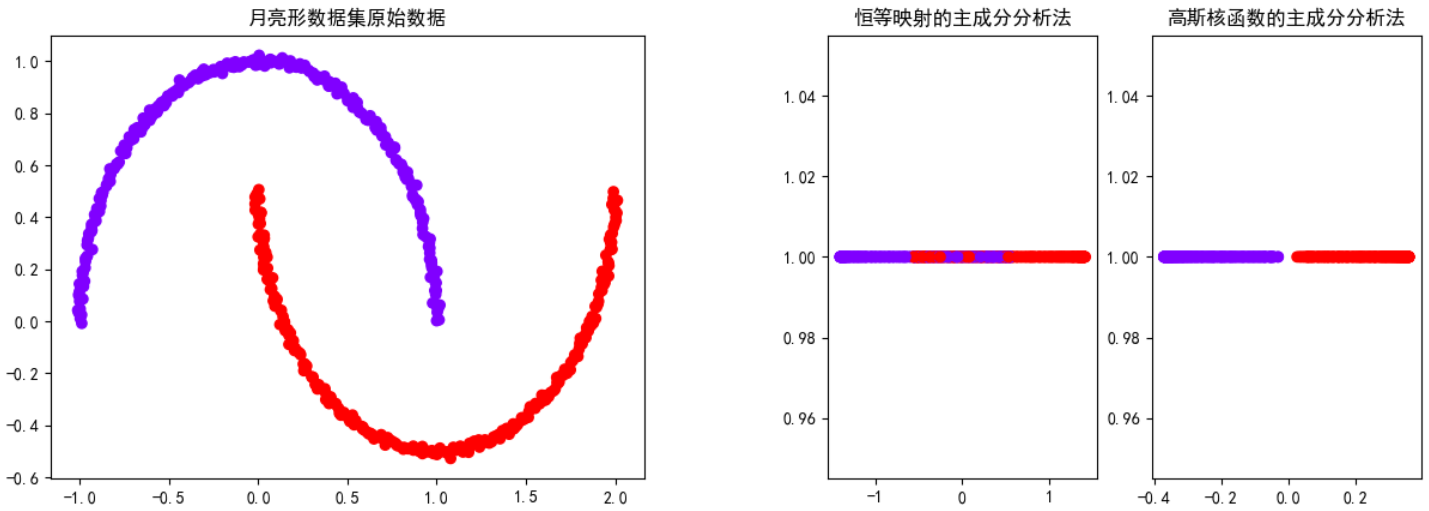
$\hat{\boldsymbol{K}} = K - JK - KJ + JKJ$

计算 $\hat{\boldsymbol{K}}$ 的特征值：$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m$

令 $\boldsymbol{u}^{(1)}, \boldsymbol{u}^{(2)}, \cdots, \boldsymbol{u}^{(m)}$ 为对应特征值的特征向量

return $Z = (\sqrt{\lambda_1}\boldsymbol{u}^{(1)}, \sqrt{\lambda_2}\boldsymbol{u}^{(2)}, \cdots, \sqrt{\lambda_d}\boldsymbol{u}^{(d)})$

主成分分析算法实现参考 kernel_pca.py
案例实战 10.2 月亮数据集的降维。moon_kernel_pca.py



(a) 月亮数据集的降维　　(b) 线性 PCA 与高斯核 PCA 对比

图 3: 月亮形数据集的 PCA 可视化对比

使用普通（线性）主成分分析法将原来平面上按颜色明确区分开的两类数据点降至一维之后，它们之间就没有明确的分界线了。而使用带核函数的主成分分析法降维之后，两种颜色的点仍然保留着明确的分界线，由此可见，在月亮形数据集上，主成分分析的核方法在降维过程中更好地保留了数据所携带的信息。因此，根据具体的问题，选择合适的核函数是非常重要的。

目录

PCA 用于对具有一组连续正交分量（Orthogonal Component）的多变量数据集进行方差最大化的分解。在 sklearn 中，PCA 被实现为一个变换器对象，通过 fit 方法可以拟合出 n 个成分，并且可以将新的数据投影到这些成分中。线性代数中的奇异值分解（Singular Value Decomposition, SVD）与主成分分析法有着紧密的联系，由奇异值分解算法可以得到主成分分析法的另一种实现方法。在应用 SVD 之前，PCA 是在为每个特征聚集而不是缩放输入数据。可选参数 whiten=True 使得可以将数据投影到奇异（singular）空间上，同时将每个成分缩放到单位方差。

案例实战 10.3 鸢尾花数据集的降维。iris_pca_sklearn.py

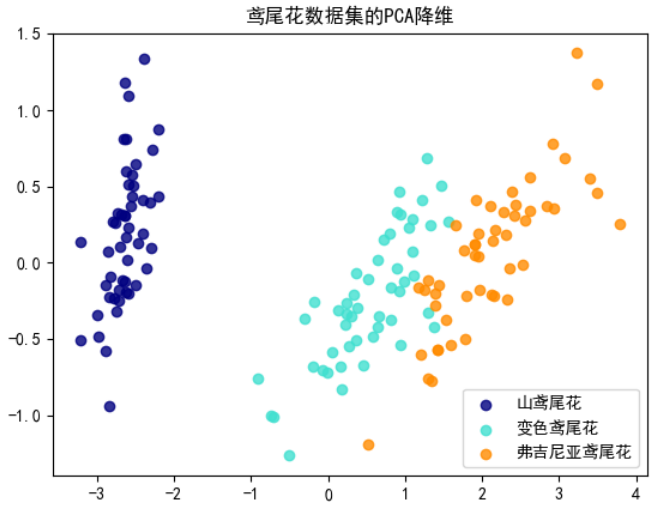


图 4: 鸢尾花数据集降维后的分类结果

鸢尾花数据集包含 4 个特征，通过 PCA 降维后投影到方差最大的二维空间上，运行该程序，得到图4，可以看出，三个种类的鸢尾花，能够很好地区分。

案例实战 10.4 环形数据的核主成分分析。参考 circles_kernel_pca_sklearn.py

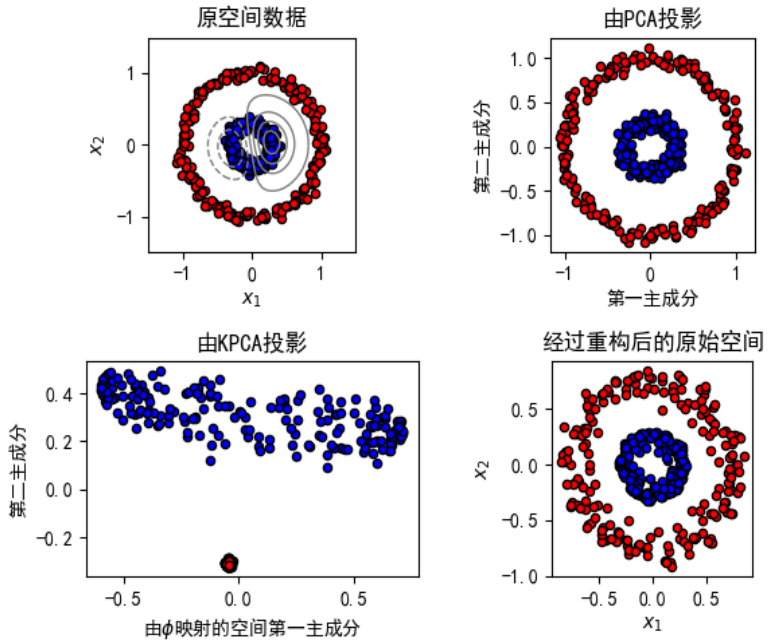
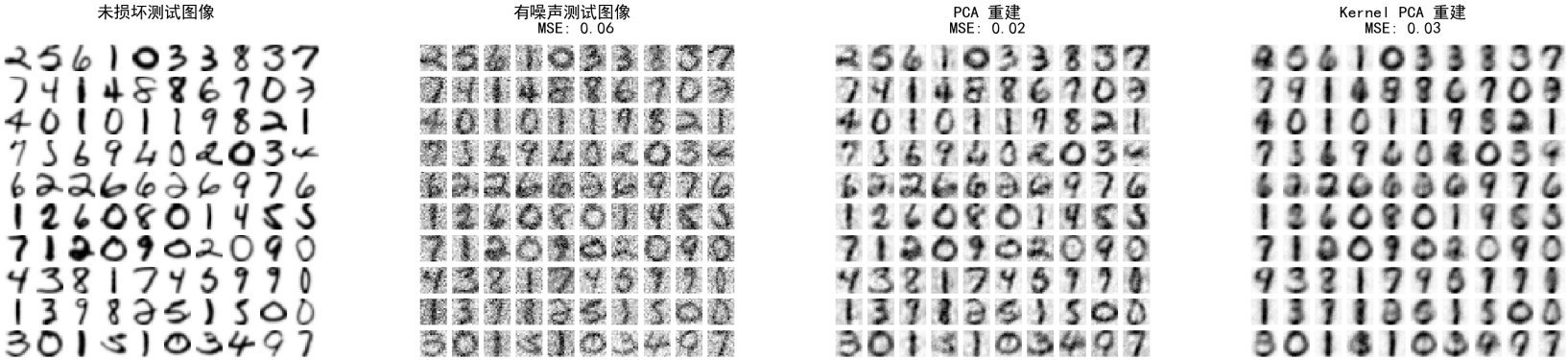


图 5: PCA 和 KPCA 投影对比

第一个子图是原始空间的数据及其不同类型的等高线。第二个子图经投影后，数据并不能线性的分离开来。第三个子图在 ϕ 空间的数据第一主成分和第二成分的图像，可以线性的将数据分离。第四个子图，数据可以很好地得到重构。此案例表明，带核主成分分析能够找到数据的投影，从而使数据线性可分。

案例实战 10.5 使用 PCA 对数字图形进行去噪。本案例展示如何使用 KernelPCA 算法进行图片去噪。就是利用从 fit 函数中学到的近似函数重建原始图片。digits_denoising_pca_sklearn.py



(a) 未损坏测试图像　(b) 有噪声测试图片　(c) PCA 重建　(d) Kernel PCA 重建

图 6: 数字图像的 PCA 降噪

第一个子图展示的是无噪声的原始数据图像。第二个子图是添加了高斯噪声之后的图像。第三个子图是使用 PCA 降噪之后的重建的图像，第四个子图则是使用 Kernel PCA 降噪之后，重建的图像。从运行的结果可以看出 PCA 的 MSE=0.02，比 Kernel PCA 的 MSE=0.1 更小，然而定性的分析并不能说明 PCA 比 Kernel PCA 更好。我们可以观察到第三和第四两个子图中，Kernel PCA 能够消除背景噪声并提供一个更为平滑的图像。此外，必须指出，Kernel PCA 算法的去噪效果依赖于参数’n_components’,’gamma’和’alpha’。读者可以调整参数进行测试。