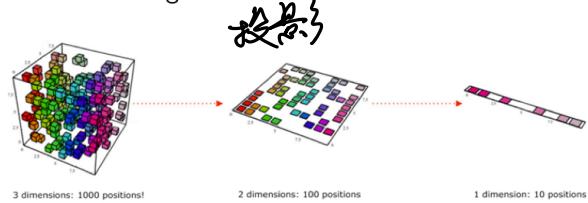


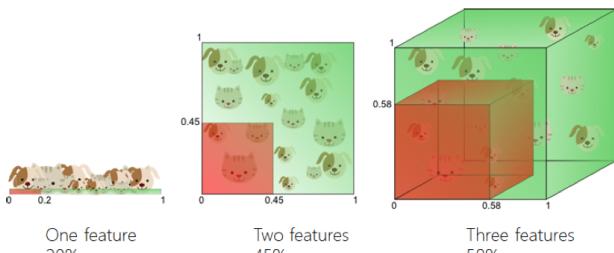
What is Dimensionality Reduction

- The process of reducing the number of random variables under consideration, via obtaining a set of “uncorrelated” principal variable
- By mapping from high-dimensional space to low-dimensional space
- Learning $f : \mathcal{X} \rightarrow \mathcal{Y}$, where $\dim \mathcal{X} = n$ and $\dim \mathcal{Y} = r$ with $n > r$.
- Including both unsupervised learning (mostly common) and supervised learning

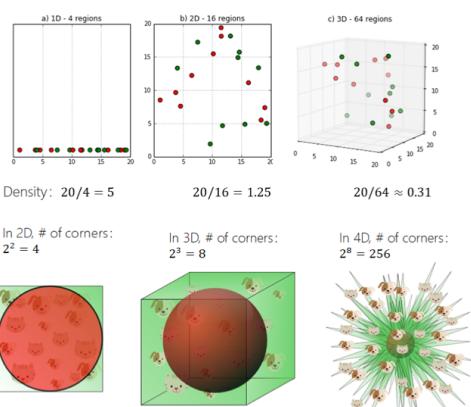


Why Need Dimensionality Reduction

- Curse of dimensionality **维数灾难**
- Eg : classify cats and dogs using features, if we want to cover 20% of the feature space, how many data do we need ?
- However, the number of samples is limited in practice

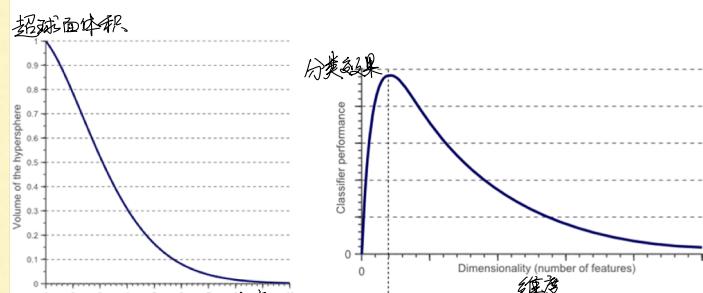


- Due to the sparsity of data in high dimensions, it is easy to overfit
- Hard to train a good model to classify the corner data (getting more in high dimensions)



Curse of Dimensionality

- The volume of hypersphere decays to zero with the increase of dimension
- The performance gets worse with the increase of dimension



将原始数据的随机变量减少
使降维后变量相关性减少

降维可以有 label, 但更多时候没有

图片、NLP、生物医学维度都能达到上千万。

①提高计算效率
②提高准确性

维数越高, 达到相同准确性所需数据量越大.

有限样本点个数, 倾向于覆盖圆形/球形
越高维越难覆盖到 feature space.

Roles of Dimensionality Reduction

- Data compression
- Denoising
- Feature extraction by mapping and feature selection (eg. Lasso)
- Reduce both spatial and time complexity, so that fewer parameters are needed and smaller computational power is required
- Data visualization

小波变换，傅立叶变换。

Methods in Dimensionality Reduction

- Linear dimensionality reduction :
 - Principal component analysis (PCA)
 - Linear discriminant analysis (LDA)
 - Independent component analysis (ICA)
- Nonlinear dimensionality reduction :
 - Kernel based methods (Kernel PCA)
 - Manifold learning (ISOMAP, Locally Linear Embedding (LLE), Multidimensional scaling (MDS), t-SNE)

ICA是PCA的一个变种，多了随机性
LDA假设数据分布高斯形
PCA和LDA都有收敛性。

Principal Component Analysis 主成分分析

Variance & Covariance Matrix

- Variance measures the variability or divergence of single variable : $\text{Var}(X) = E(X - EX)^2$, sample version $S^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$; standard deviation : $\text{Std}(X) = \sqrt{\text{Var}(X)}$
- For more variables, $\text{Cov}(X, Y) = E(X - EX)(Y - EY)$, sample version $C = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$
- If $X = (x_1, \dots, x_n)^T \in \mathbb{R}^{n \times p}$ is the sample matrix, then $C = \frac{1}{n-1} (X - \mathbf{1}_n \bar{x}^T)^T (X - \mathbf{1}_n \bar{x}^T) = \frac{1}{n-1} (X - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^T X)^T (X - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^T X) = \frac{1}{n-1} X^T J X$, where $J = I_n - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^T$ is a projection matrix with rank $n-1$.

无监督，线性降维。

用正交变换将强相关转成弱相关。

新变量称为主成分。

可以用方差的大小刻画信息量的多少。

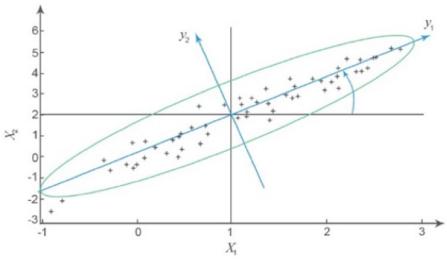
新变量不是原始变量但有关系。

Principal Component Analysis (PCA)

- PCA transforms a set of strongly correlated variables to another set (**typically much smaller**) of **weakly correlated** variables by using orthogonal transformation
- The new variables are called principal components
- The new set of variables are linear combinations of the original variables whose variance information is inherited as much as possible
- Unsupervised learning
- Proposed by Karl Pearson, successfully used in economics by Stone (1947) : keep 97.4% information, 17 variables about income and expenditure are finally reduced to 3 variables (F1 : total income, F2 : rate of change in total income, F3 : economic development or recession)

Geometric Interpretation

- Assume a set of 2D data follows Gaussian distribution (but not limited to Gaussian distribution!), the reduction to 1D is successfully achieved by taking a direction with larger variance (larger variability of data)
- The direction in the major axis contains more information than the other direction, since smaller variance indicates the variables are almost the same



坐标线性组合.

Linear Algebra

- Let $\{\mathbf{e}_i\}_{i=1}^p$ be the canonical basis in Euclidean space, want to find another orthonormal basis $\{\tilde{\mathbf{e}}_i\}_{i=1}^p$ such that the random vector $\mathbf{v} = \sum_{i=1}^p x_i \mathbf{e}_i$ can be expressed in the new basis by $\mathbf{v} = \sum_{i=1}^p \tilde{x}_i \tilde{\mathbf{e}}_i$, where $\text{Var}(\tilde{x}_1) \geq \dots \geq \text{Var}(\tilde{x}_p)$ and $\text{Cov}(\tilde{x}_i, \tilde{x}_j) \approx 0$ for $i \neq j$
- By linear algebra, the coordinate transformation is given by the linear transformation: $(\tilde{\mathbf{e}}_1, \dots, \tilde{\mathbf{e}}_p) = (\mathbf{e}_1, \dots, \mathbf{e}_p)W$, where $W \in \mathbb{R}^{p \times p}$ is an invertible matrix
- The component coefficients is transformed accordingly :
 $\mathbf{x} = W\tilde{\mathbf{x}}$

Eigendecomposition of Sample Covariance Matrix

- Assume we have n centralized samples $\{\mathbf{x}_i\}_{i=1}^n$ with $\frac{1}{n} \sum_{i=1}^n \mathbf{x}_i = \mathbf{0}_p$
- Then $X^T = (\mathbf{x}_1, \dots, \mathbf{x}_n) = W(\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_n) = W\tilde{X}^T$
- The sample covariance matrix of X is $\text{Cov}(X) = \frac{1}{n-1} X^T X$
- The sample covariance matrix of \tilde{X} is $\text{Cov}(\tilde{X}) = \frac{1}{n-1} \tilde{X}^T \tilde{X} = \frac{1}{n-1} W^T X^T X W = W^T \text{Cov}(X) W$
- Its diagonals are the sample versions of $\text{Var}(\tilde{x}_1), \dots, \text{Var}(\tilde{x}_p)$, while its off-diagonals are the covariances between \tilde{x}_i and \tilde{x}_j
- Need that $\text{Cov}(\tilde{X})$ is nearly diagonal with decreasing diagonal entries for some W .
- Equivalent to do eigendecomposition :
 $\text{Cov}(X) = O \text{diag}(\lambda_1, \dots, \lambda_p) O^T$ with some orthogonal matrix $O \in \mathbb{R}^{p \times p}$ and $\lambda_1 \geq \dots \geq \lambda_p \geq 0$, then let $W = O$ completes the job

中心化 \leftrightarrow 平移 \leftrightarrow 令均值为0

相合变化

对称半正定

对原始协方差矩阵做特征值分解

方差最大化

λ_i 是非负的

r: 主成分个数

$\tilde{\mathbf{w}}_1, \dots, \tilde{\mathbf{w}}_r$: 新坐标系的基。

第 r 个主成分即 $\tilde{\mathbf{w}}_r^T \mathbf{x}$

Interpretations

- Variances in the transformed variables : $\text{Var}(\tilde{x}_i) = \lambda_i$, eigenvalues of $\text{Cov}(X)$
- The new basis consists of the columns of $W = O$, i.e., the eigenvectors of $\text{Cov}(X)$
- The percentage $\frac{\lambda_i}{\sum_{j=1}^p \lambda_j}$ explains the importance of the new variable \tilde{x}_i
- Given a threshold t , we can choose the number of variables r such that the total contribution to the variance of the new r variables $\sum_{i=1}^r \frac{\lambda_i}{\sum_{j=1}^p \lambda_j}$ exceeds the threshold t . Thus these r directions $\mathbf{w}_1, \dots, \mathbf{w}_r$ are enough to represent the original n variables
- For any random vector $\mathbf{x} \in \mathbb{R}^p$, the corresponding r principal components are thus $\mathbf{w}_1^T \mathbf{x}, \dots, \mathbf{w}_r^T \mathbf{x}$

Another Viewpoint - Best Reconstruction 最优重构

- Note that the new basis $\{\tilde{\mathbf{e}}_j\}_{j=1}^p$ is given by $\tilde{\mathbf{e}}_j = \mathbf{w}_j$;
- After the projection (if we keep the first r components), the projected point of each sample \mathbf{x}_i is $\tilde{\mathbf{x}}_{i,1}\mathbf{w}_1 + \dots + \tilde{\mathbf{x}}_{i,r}\mathbf{w}_r$, where the coordinate is given by $\tilde{\mathbf{x}}_{i,j} = \mathbf{w}_j^T \mathbf{x}_i$;
- The reconstruction error is the sum of all squared L^2 errors of all samples :

$$\begin{aligned} RE(W) &= \sum_{i=1}^n \left\| \sum_{j=1}^r \tilde{\mathbf{x}}_{i,j} \mathbf{w}_j - \mathbf{x}_i \right\|_2^2 = \sum_{i=1}^n \|(\mathbf{W}_r \mathbf{W}_r^T - I) \mathbf{x}_i\|_2^2 \\ &= \sum_{i=1}^n \mathbf{x}_i^T (I - \mathbf{W}_r \mathbf{W}_r^T) \mathbf{x}_i = \text{Tr} \left(\sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^T (I - \mathbf{W}_r \mathbf{W}_r^T) \right) \\ &= \text{Tr} (\mathbf{X}^T \mathbf{X} (I - \mathbf{W}_r \mathbf{W}_r^T)) = \text{Tr} (\mathbf{X}^T \mathbf{X}) - \text{Tr} (\mathbf{W}_r^T \mathbf{X}^T \mathbf{X} \mathbf{W}_r) \end{aligned}$$

- Resulting in an optimization problem :

$$\min_{\mathbf{W}_r} -\text{Tr} (\mathbf{W}_r^T \mathbf{X}^T \mathbf{X} \mathbf{W}_r), \quad \text{subject to } \mathbf{W}_r^T \mathbf{W}_r = I$$

PCA Algorithm

- Given the data matrix $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^T \in \mathbb{R}^{n \times p}$ and a threshold t (in some other cases, the number of principal components r) :
 - Centralize the data by their mean $\bar{\mathbf{x}} = \frac{1}{n} \mathbf{1}_n^T \mathbf{X}$, and compute the sample covariance matrix $C = \frac{1}{n-1} (\mathbf{X} - \mathbf{1}_n \bar{\mathbf{x}}^T)^T (\mathbf{X} - \mathbf{1}_n \bar{\mathbf{x}}^T)$
 - Compute the eigenvalues $\{\lambda_i\}_{i=1}^p$ and the corresponding eigenvectors $\{\mathbf{w}_i\}_{i=1}^p$
 - Order the eigenvalues as $\lambda_{(1)} \geq \dots \geq \lambda_{(p)}$, and compose an orthogonal matrix W by the eigenvectors columnwise in the same order : $W = (\mathbf{w}_1, \dots, \mathbf{w}_p)$
 - Compute the variance contribution of the first r eigenvalues : $\sum_{i=1}^r \frac{\lambda_i}{\sum_{j=1}^p \lambda_j}$, find a suitable r such that this variance contribution is greater than the threshold t
 - Pick the first r columns in W and form a matrix $\mathbf{W}_r = (\mathbf{w}_1, \dots, \mathbf{w}_r) \in \mathbb{R}^{p \times r}$
 - Output $\tilde{\mathbf{X}}_r = \mathbf{X} \mathbf{W}_r \in \mathbb{R}^{n \times r}$ as the projected data matrix, whose rows consist of data points in r dimensional subspace

An Example

- The data : the monthly prices of three brands of vehicles (Jeep : x_1 , Toyota : x_2 , Benz : x_3)
- The covariance matrix is given by

$$C = \begin{pmatrix} 1 & \frac{2}{\sqrt{10}} & -\frac{2}{\sqrt{10}} \\ \frac{2}{\sqrt{10}} & 1 & -\frac{4}{5} \\ -\frac{2}{\sqrt{10}} & -\frac{4}{5} & 1 \end{pmatrix}$$

- Compute the characteristic polynomial :

$$\det(\lambda I - C) = \begin{vmatrix} \lambda - 1 & -\frac{2}{\sqrt{10}} & \frac{2}{\sqrt{10}} \\ -\frac{2}{\sqrt{10}} & \lambda - 1 & \frac{4}{5} \\ \frac{2}{\sqrt{10}} & \frac{4}{5} & \lambda - 1 \end{vmatrix}$$

- Solve for the eigenvalues : $\lambda_1 = 2.38$, $\lambda_2 = 0.42$, $\lambda_3 = 0.2$

使重构误差最小化

$$\begin{aligned} &\sum_i \mathbf{x}_i^T (I - \mathbf{W}_r \mathbf{W}_r^T) \mathbf{x}_i \\ &= -\text{Tr} (\mathbf{x}_i \mathbf{x}_i^T (I - \mathbf{W}_r \mathbf{W}_r^T)) \\ &= \text{Tr} \left(\sum_i \mathbf{x}_i \mathbf{x}_i^T (I - \mathbf{W}_r \mathbf{W}_r^T) \right) \\ &= \text{Tr} (\mathbf{X}^T \mathbf{X} (I - \mathbf{W}_r \mathbf{W}_r^T)) \\ &\quad \mathbf{W}_r \mathbf{W}_r^T - I 是投影矩阵. = \text{Tr} (\mathbf{X}^T \mathbf{X}) - \text{Tr} (\mathbf{X}^T \mathbf{X} \mathbf{W}_r \mathbf{W}_r^T) \\ &= \text{Tr} (\mathbf{X}^T \mathbf{X}) - \text{Tr} (\mathbf{W}_r^T \mathbf{X}^T \mathbf{X} \mathbf{W}_r) \end{aligned}$$

→ 相当于矩阵特征值分解.

$$\mathbf{X} = \begin{pmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{pmatrix} \quad \bar{\mathbf{x}} = (2 \ 2)$$

$$\mathbf{C} = \frac{1}{2-1} \begin{pmatrix} -1 & -1 \\ 0 & 0 \\ 1 & 1 \end{pmatrix}^T \begin{pmatrix} -1 & -1 \\ 0 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}$$

- Plug in each eigenvalues and solve for the corresponding eigenvectors, e.g., $(\lambda_1 I - C) \mathbf{w}_1 = \mathbf{0}$, or equivalently,

$$\begin{cases} 1.38w_{11} - \frac{2}{\sqrt{10}}w_{12} + \frac{2}{\sqrt{10}}w_{13} = 0, \\ -\frac{2}{\sqrt{10}}w_{11} + 1.38w_{12} + 0.8w_{13} = 0, \\ \frac{2}{\sqrt{10}}w_{11} + 0.8w_{12} + 1.38w_{13} = 0. \end{cases}$$

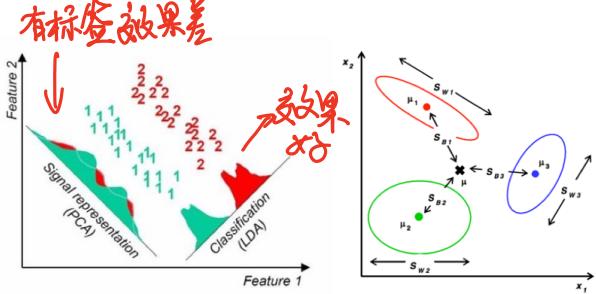
- One can find three eigenvectors as $\mathbf{w}_1 = (0.54, 0.59, -0.59)^T$, $\mathbf{w}_2 = (0.84, -0.39, 0.39)^T$, $\mathbf{w}_3 = (0, 0.71, 0.71)^T$
- The three components are

$$\begin{aligned} \tilde{x}_1 &= \mathbf{w}_1^T \mathbf{x} = 0.54x_1 + 0.59x_2 - 0.59x_3, \\ \tilde{x}_2 &= \mathbf{w}_2^T \mathbf{x} = 0.84x_1 - 0.39x_2 + 0.39x_3, \\ \tilde{x}_3 &= \mathbf{w}_3^T \mathbf{x} = 0.71x_2 + 0.71x_3. \end{aligned}$$

- As $\lambda_1 \gg \lambda_2, \lambda_3$, the first principal component \tilde{x}_1 reflects the change of prices in all three brands of vehicles

Linear Discriminant Analysis (LDA)

- Supervised learning : based on the labels, do linear projection in order to maximize the between-class point scatter (variability) in low dimensions
- Initially proposed by R. Fisher for two-class classification (1936)
- Generalized by C. R. Rao (1948) to K classes $\{C_1, \dots, C_K\}$



Basic Concepts

- The number of samples in each class is $n_k = \sum_{i: x_i \in C_k} 1$, whereas the total number of samples is $n = \sum_{k=1}^K n_k$
- The mean of samples in class k is $\mu_k = \frac{1}{n_k} \sum_{i: x_i \in C_k} x_i$, whereas the mean of all samples is $\mu = \sum_{k=1}^K \frac{n_k}{n} \mu_k$
- Before projection, the between-class point scatter is $S_b = \sum_{k=1}^K \frac{n_k}{n} (\mu_k - \mu)(\mu_k - \mu)^T$; after projection $W_r \in \mathbb{R}^{p \times r}$, the between-class point scatter is $\tilde{S}_b = W_r^T S_b W_r$
- Before projection, the within-class point scatter (variance) for each class C_k is $S_k = \frac{1}{n_k} \sum_{i: x_i \in C_k} (x_i - \mu_k)(x_i - \mu_k)^T$, thus the total within-class point scatter is $S_w = \sum_{k=1}^K \frac{n_k}{n} S_k$; after projection, the within-class point scatter for each class C_k is $\tilde{S}_k = W_r^T S_k W_r$, and the total within-class point scatter is $\tilde{S}_w = W_r^T S_w W_r$

Optimization Problem

- Need to find the optimal directions (columns of W_r) such that the between-class point scatter \tilde{S}_b is maximized and within-class point scatter \tilde{S}_w is minimized, i.e.,

$$\max_{\mathbf{w}} J(\mathbf{w}) = \frac{\mathbf{w}^T S_b \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}}$$

- This is equivalent to solve

$$\max_{\mathbf{w}} J_b(\mathbf{w}) = \mathbf{w}^T S_b \mathbf{w}, \quad \text{subject to } \mathbf{w}^T S_w \mathbf{w} = 1$$

- By introducing a Lagrange multiplier λ , we define Lagrangian as $L(\mathbf{w}, \lambda) = \mathbf{w}^T S_b \mathbf{w} - \lambda(\mathbf{w}^T S_w \mathbf{w} - 1)$
- The optima is obtained as the solution to the equation

$$\nabla_{\mathbf{w}} L = 2S_b - 2\lambda S_w \mathbf{w} = \mathbf{0} \Rightarrow S_w^{-1} S_b \mathbf{w} = \lambda \mathbf{w}$$

- The optimal directions are the eigenvectors of $S_w^{-1} S_b$

只有高斯时 PCA 和 LDA 解释方法。

中类, k 分得越开越好
内部点越靠近越好。

类间协方差矩阵

类内协方差矩阵

类内散度的逆乘类间散度

An Example

- Given two sets of data : class 1 is $\{(4, 1)^T, (2, 4)^T, (2, 3)^T, (3, 6)^T, (4, 4)^T\}$, and class 2 is $\{(9, 10)^T, (6, 8)^T, (9, 3)^T, (8, 7)^T, (10, 8)^T\}$
- Class means : $\mu_1 = (3, 3.6)^T$, $\mu_2 = (8.4, 7.6)^T$, the point scatter metrics are

$$S_1 = \begin{pmatrix} 0.8 & -0.4 \\ -0.4 & 2.6 \end{pmatrix}, \quad S_2 = \begin{pmatrix} 1.84 & -0.28 \\ -0.28 & 5.36 \end{pmatrix},$$

$$S_b = \begin{pmatrix} 7.29 & 4.86 \\ 4.86 & 3.24 \end{pmatrix}, \quad S_w = \begin{pmatrix} 1.32 & -0.34 \\ -0.34 & 4 \end{pmatrix}.$$

- The eigenvalue of $S_w^{-1}S_b$ is solved from

$$0 = \det(\lambda I - S_w^{-1}S_b) = \begin{vmatrix} \lambda - 5.97 & -3.98 \\ -1.72 & \lambda - 1.15 \end{vmatrix} \Rightarrow \lambda = 7.11$$

PCA vs. LDA

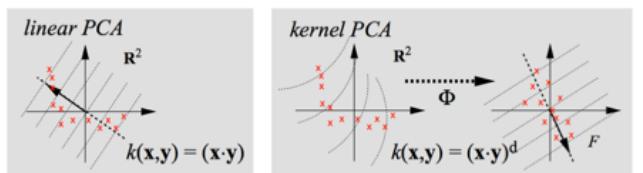
- PCA
 - Start from sample covariance matrix and find directions with maximal variances
 - Unsupervised learning, used as pre-training step, must be coupled with other learning methods
- LDA
 - Make use of labels and find projections after which the classification becomes more obvious
 - Supervised learning, can be used as classification or coupled with other learning methods

两个都用了特征值分解方法

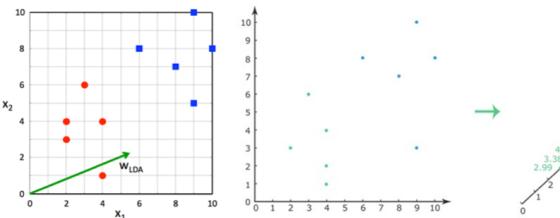
Non Linear Dimensionality Reduction

Kernel PCA

- PCA works well for Gaussian distribution
- If the data do not follow Gaussian, we can find a map $\phi: \mathbb{R}^p \rightarrow \mathbb{R}^q$ so that $\phi(x)$ (almost) follows Gaussian
- We can do PCA for the transformed data $\{\phi(x_i)\}_{i=1}^n$
- Similar to nonlinear SVM, kernel trick can be used to avoid explicit computation of ϕ



- The optimal directions is $w^* = (0.96, 0.28)^T$
- After projection, the data become 1D :
 - Class 1 : $\{4.12, 3.03, 2.75, 4.55, 4.95\}$
 - Class 2 : $\{11.42, 7.98, 9.48, 9.63, 11.83\}$



无监督学习，常用于预训练，必须与其他学习方法一起用。
LDA将降维和分类同时做
有监督学习，可用于分类或其他学习方法中。

PCA在高斯分布下表现好。
核函数是反对称半正定的。

经核函数处理不是为了升维，而是为了改变分布。

Covariance Matrix in Transformed Space

- Assume the transformed data are centralized :
 $\mu = \frac{1}{n} \sum_{i=1}^n \phi(\mathbf{x}_i) = 0$
- Covariance Matrix $\tilde{C} = \frac{1}{n-1} \sum_{i=1}^n \phi(\mathbf{x}_i) \phi(\mathbf{x}_i)^T$
- Do PCA for transformed data is equivalent to find the eigenvalues and eigenvectors of \tilde{C}
- Let λ be an eigenvalue of \tilde{C} and $\mathbf{v} \in \mathbb{R}^q$ be the corresponding eigenvector, i.e., $\tilde{C}\mathbf{v} = \lambda\mathbf{v}$.
- It can be shown that $\mathbf{v} = \sum_{i=1}^n \alpha_i \phi(\mathbf{x}_i)$ where $\alpha_i = \frac{1}{\lambda(n-1)} \phi(\mathbf{x}_i)^T \mathbf{v}$
- Furthermore, $\alpha_i = \frac{1}{\lambda(n-1)} \sum_{j=1}^n K(\mathbf{x}_i, \mathbf{x}_j) \alpha_j$, where $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ is kernel function
- It is sufficient to solve the eigenvalue problem :
 $K\alpha = \lambda(n-1)\alpha$ where $K = (K(\mathbf{x}_i, \mathbf{x}_j))_{i,j}$ is kernel matrix and $\alpha = (\alpha_i)$ is the coefficient vector of \mathbf{v}

Kernel PCA Algorithm

- Choose a kernel function $K(x, y)$ satisfying the necessary properties
- Compute the kernel matrix $K = (K(\mathbf{x}_i, \mathbf{x}_j))_{i,j}$
- Compute the eigenvalues $\lambda_1 \geq \dots \geq \lambda_q$ and eigenvectors $\alpha^{(1)}, \dots, \alpha^{(q)}$ of K
- For any new sample \mathbf{x} , the j component after projection is

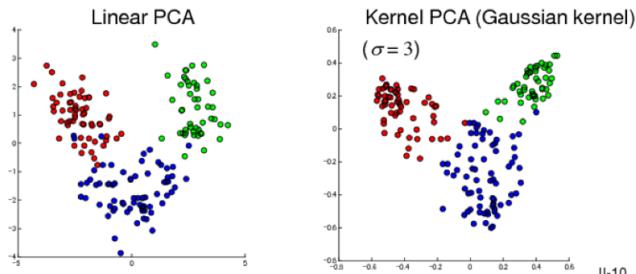
$$z_j = \mathbf{v}_j^T \phi(\mathbf{x}) = \sum_{i=1}^n \alpha_i^{(j)} K(\mathbf{x}_i, \mathbf{x})$$

An Example

Wine data (from UCI repository)

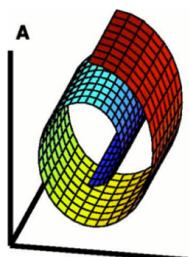
13 dim. chemical measurements of for three types of wine. 178 data. Class labels are NOT used in PCA, but shown in the figures.

First two principal components:



Manifold Learning 流形学习

- A manifold is a topological space that locally resembles Euclidean space near each point. It generalizes the concepts of curves and surfaces in Euclidean space.
- The dimension of a manifold is the minimal number of coordinates to represent a point on the manifold
- Some dimensionality reduction methods are based on the concept of manifold : ISOMAP, LLE, MDS, t-SNE



对转化后的数据做PCA.

$$\begin{aligned} \tilde{C}\vec{v} = \lambda \vec{v} &\Leftrightarrow \frac{1}{n-1} \sum_{i=1}^n \phi(\mathbf{x}_i) \phi(\mathbf{x}_i)^T \vec{v} = \vec{v} \\ \Rightarrow \vec{v} &= \sum_{i=1}^n \alpha_i \phi(\mathbf{x}_i), \quad \alpha_i = \frac{1}{\lambda(n-1)} \phi(\mathbf{x}_i)^T \vec{v} \\ \Rightarrow \alpha_i &= \frac{1}{\lambda(n-1)} \phi(\mathbf{x}_i)^T \sum_{j=1}^n \alpha_j \phi(\mathbf{x}_j) = \frac{1}{\lambda(n-1)} \sum_{j=1}^n K(\mathbf{x}_i, \mathbf{x}_j) \alpha_j \\ \Rightarrow \sum_{i=1}^n \lambda(n-1) \alpha_i &= \sum_{i=1}^n \sum_{j=1}^n K(\mathbf{x}_i, \mathbf{x}_j) \alpha_j \quad (\text{核函数 } K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)) \\ \Rightarrow \lambda(n-1) \alpha &= K\alpha \end{aligned}$$

在 $(\phi(\mathbf{x}_i))_{i=1}^n$ 中选取主成分 $k = 1 \dots r$

$$\mathbf{v}^{(k)} = \sum_i \alpha_i^{(k)} \phi(\mathbf{x}_i)$$

$$\phi(\mathbf{x})^T \mathbf{v}^{(k)} = \sum_i \alpha_i^{(k)} \phi(\mathbf{x})^T \phi(\mathbf{x}_i) = \sum_i \alpha_i^{(k)} K(\mathbf{x}, \mathbf{x}_i)$$

对 $K = (K(\mathbf{x}_i, \mathbf{x}_j))_{i,j}$ 做特征值分解 $K\alpha^{(k)} = \lambda^{(k)} \alpha^{(k)}$

$$K = (K(\mathbf{x}_i, \mathbf{x}_j)) = (\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)) = \begin{pmatrix} (\phi(\mathbf{x}_1))^T \\ \vdots \\ (\phi(\mathbf{x}_n))^T \end{pmatrix} (\phi(\mathbf{x}_1) \dots \phi(\mathbf{x}_n))$$

$$= \Phi(\mathbf{x}) \Phi(\mathbf{x})^T = AA^T \in \mathbb{R}^{n \times n} \quad (A \in \mathbb{R}^{n \times q})$$

$$(n-1)\tilde{C} = \sum_{i=1}^n \phi(\mathbf{x}_i) \phi(\mathbf{x}_i)^T = \Phi(\mathbf{x})^T \Phi(\mathbf{x}) = A^T A \in \mathbb{R}^{q \times q}$$

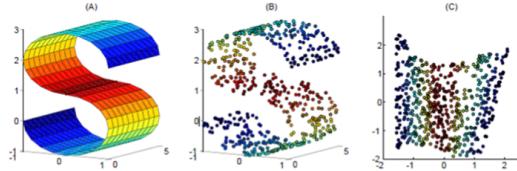
K 的特征值 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_q \geq 0 = 0 \dots = 0$ 取前 q 个。

将高维空间中的曲面拉直成平面

无监督

Locally Linear Embedding (LLE) 局部线性嵌入

- Reduce the number of free coordinates while keeping the local geometric structure of the data, e.g., if \mathbf{x}_A and \mathbf{x}_B are neighbor in high dimension, after the dimension reduction (transformation), they must be close to each other in low dimension
- The clustering effect should also be inherited



LLE Reconstruction

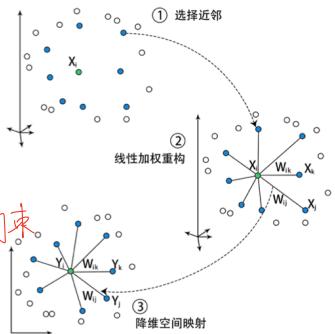
- Assume each data point is locally linearly dependent of its neighbors : it can be written as the linear combination of its K nearest neighbors $\{\mathbf{x}_{k_j}\}_{j=1}^K$, with the KNN indices $\{k_j\}_{j=1}^K$

- The weight is determined by the optimization for each \mathbf{x}_i :

$$\min_{\mathbf{w}} \|\mathbf{x}_i - \sum_{j=1}^K w_{ik_j} \mathbf{x}_{k_j}\|_2^2$$

subject to $\sum_{j=1}^K w_{ik_j} = 1, \quad w_{ij} \geq 0$

where $w_{ij} = 0$ if $j \notin \{k_j\}_{j=1}^K$



Low Dimensional Representation

- In r ($r < p$) dimensional space, find n points such that the local structure (e.g., clustering effect) is preserved

$$\min_{\mathbf{y}_1, \dots, \mathbf{y}_n} \sum_{i=1}^n \left\| \mathbf{y}_i - \sum_{j=1}^n w_{ij} \mathbf{y}_j \right\|_2^2 \xrightarrow{\text{(W)} \begin{pmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_n \end{pmatrix} \text{只取第 } i \text{ 行}} = \sum_i \|[\mathbf{I} - \mathbf{W}] \mathbf{y}_i\|_2^2 \xrightarrow{\text{A}} = \sum_i \|\mathbf{A}_i\|_2^2$$

$$= \sum_i \mathbf{A}_i^T \mathbf{A}_i$$

$$= \text{Tr}(\mathbf{A}^T \mathbf{A})$$

$$= \text{Tr} \underbrace{[\mathbf{Y} (\mathbf{I} - \mathbf{W})^T (\mathbf{I} - \mathbf{W}) \mathbf{Y}^T]}_{\mathbf{M}} = \text{Tr}(\mathbf{Y}^T \mathbf{M} \mathbf{Y})$$

- This is equivalent to the matrix minimization problem

$$\min_{\mathbf{Y}} \text{Tr}(\mathbf{Y}^T \mathbf{M} \mathbf{Y}), \quad \text{s.t. } \mathbf{Y} \mathbf{Y}^T = \mathbf{I}, \quad \xrightarrow{\text{与谱聚类类似}}$$

where $\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_n)^T \in \mathbb{R}^{n \times r}$ and $\mathbf{M} = (\mathbf{I} - \mathbf{W})(\mathbf{I} - \mathbf{W})^T$ with $\mathbf{W} = (w_{ij})_{i,j=1}^n$ being the weight matrix (not necessarily symmetric)

- This is solved by eigen-decomposition : The columns of \mathbf{Y} consist of the r eigenvectors corresponding to the r smallest eigenvalues of \mathbf{M}

Summary of LLE

- Only one tuning parameter K
- Linear algebra computation
- Only local information, no global information
- No explicit mapping as in PCA ($\tilde{\mathbf{X}}_r = \mathbf{X} \mathbf{W}_r$)

只有参数 K 需要调，即邻居样本数。
(正则化系数也需调)

没有如PCA那样有显性的映射。

Multidimensional Scaling (MDS)

- For data points in high dimensional space, $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^P$, find the distance or dissimilarity matrix $\{d_{ij}\}_{i,j}^n$, e.g., $d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$
- Find $\{\mathbf{y}_i\}_{i=1}^n \subset \mathbb{R}^r$ ($r < p$), such that the distance information is preserved :

$$\min_{\mathbf{y}_1, \dots, \mathbf{y}_n} S_M(\mathbf{y}_1, \dots, \mathbf{y}_n)$$

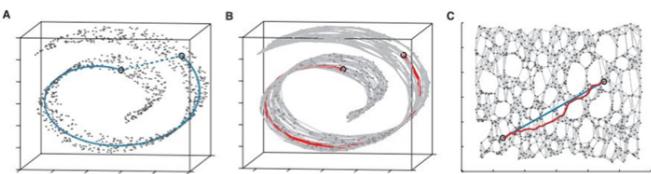
尽量使低维距离与高维
距离相近

where $S_M(\mathbf{y}_1, \dots, \mathbf{y}_n) = \sum_{i \neq j} (d_{ij} - \|\mathbf{y}_i - \mathbf{y}_j\|)^2$ is the stress function. This is called least square or Kruskal-Shephard scaling

- Alternative objective function (Sammon mapping) : $S_{S_M}(\mathbf{y}_1, \dots, \mathbf{y}_n) = \sum_{i \neq j} \frac{(d_{ij} - \|\mathbf{y}_i - \mathbf{y}_j\|)^2}{d_{ij}}$ takes care of small d_{ij}
- This is nonconvex minimization

The Motivation of ISOMAP

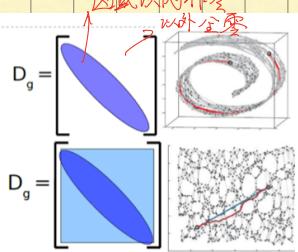
- The distance between two points may be different in different metrics (manifold metric vs. Euclidean metric)
- Geodesic distance could be a good metric instead of Euclidean distance
- Computation of geodesic distance, minimal path in graph



ISOMAP Algorithm

- Construct KNN graph $G = (V, E)$:
 - For each \mathbf{x}_i , find its K nearest neighbors $\{\mathbf{x}_j\}_{j \in N(i)}$
 - The weight of the edge $< i, j >$ between \mathbf{x}_i and \mathbf{x}_j is the Euclidean distance for each $j \in N(i)$
 - Use Floyd algorithm to compute the minimal path between each pair of vertices (i, j) as the geodesic distance $d_G(i, j)$
 - Find the low dimensional representation (e.g. by MDS) :

$$\min_{\mathbf{y}_1, \dots, \mathbf{y}_n} \sum_{i \neq j} (d_G(i, j) - \|\mathbf{y}_i - \mathbf{y}_j\|)^2$$



尽量使高维和低维中点对的距离一样.

Forbinie's norm

$$\|A - B\|_F^2 = \sum_{i,j} 1 (A - B)_{i,j})^2$$

非线性非凸，难解

距离应该是曲面上的最短距离.
不能是欧氏距离.

先构造KNN图

用两点间距离作为边的权值.

$d_G(i, j)$ 是高维图中点对的曲面距离.

Floyd Algorithm (Complexity $O(n^3)$)

1. Initialization :

$$d_G(i, j) = \begin{cases} d_x(i, j), & \text{if } < i, j > \in E \\ \infty, & \text{otherwise} \end{cases}$$

- For each pair (i, j) , update the distance as follows : for each $k = 1, \dots, n$, $d_G(i, j) = \min\{d_G(i, j), d_G(i, k) + d_G(k, j)\}$
- The final output $d_G(i, j)$ is the geodesic distance between i and j

Summary of ISOMAP

- Only one tuning parameter K
- High computational power
- Preserve the global information
- Sensitive to noise

ISOMAP是基于MDS改进的降维方法

对噪声敏感

t -distributed Stochastic Neighbor Embedding (t-SNE)

- Developed by Laurens van der Maaten and Geoffrey Hinton
- Effective for data visualization in 2D and 3D, applications in computer security research, music analysis, cancer research, especially for bioinformatic data
- Often display clusters in low dimensional space (may be false findings)
- With special parameter choices, approximates a simple form of spectral clustering

在高维中远的在低维中会更远。
在高维中近的在低维中会更近。
但降维得太厉害有时效果不好。
有凝聚类的效果，能同时做降维和分类。

Similarity in High Dimensional Space

- For data points in high dimensional space, $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^P$, find the similarity of \mathbf{x}_i and \mathbf{x}_j in the form of probability p_{ij}
- The similarity of data point x_j to data point x_i is the conditional probability, $p_{j|i}$, that x_i would pick x_j as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at x_i :

x_i 到 x_j 的概率

$$p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2 / 2\sigma_i^2)}$$

- $p_{ij} = (p_{j|i} + p_{i|j})/2n$, $p_{ii} = 0$
- The bandwidth is adapted to the density of the data : smaller values of σ_i are used in denser parts of the data space

稠密的地方用更小的 σ .

Similarity in Low Dimensional Space

- t-SNE aims to learn a set of low dimensional data $\mathbf{y}_1, \dots, \mathbf{y}_n \in \mathbb{R}^r$ that reflects the similarity p_{ij} as well as possible
- The similarity between the data point \mathbf{y}_i and \mathbf{y}_j follows t-distribution : (assume $q_{ii} = 0$)

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq i} (1 + \|\mathbf{y}_k - \mathbf{y}_i\|^2)^{-1}}$$

- t-distribution is heavy tailed so that large p_{ij} (dissimilar data pair) leads to even larger q_{ij} (falls apart)
- The closedness between the two similarity measures p_{ij} and q_{ij} is given by the Kullback-Leibler divergence :

$$D_{KL}(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

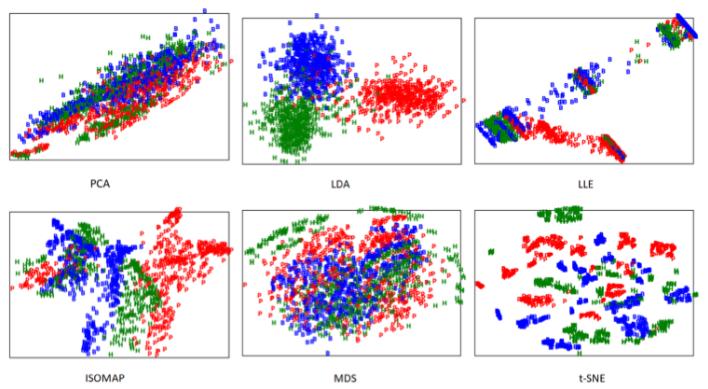
小的 p_{ij} 会引出更大的 q_{ij} ；大的会引出更小的，但更关注变小的影响).

衡量 P_{ij} 和 q_{ij} 的相似度可用 KL 散度.

t-SNE 是 MDS 的一个推广。

t-SNE 视化好，但是结果不一定对。

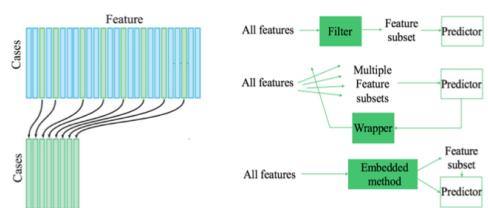
Comparison | Optical Character Recognition



Feature Selection

What is Feature Selection

- Low computational cost, better accuracy (avoid overfitting), and better interpretation,
- Feature engineering : feature extraction and selection. Feature extraction is according to the knowledge of the professions, usually done by expertise in the professional areas
- Three types : Filter, Wrapper, and Embedded



Subset Selection

- Subset search :
 - Forward search (forward stepwise, forward stagewise) :
 $\emptyset \Rightarrow \{x_1\} \Rightarrow \{x_1, x_4\} \Rightarrow \dots$
 - Backward search (backward stepwise) :
 $\{x_1, x_2, \dots, x_p\} \Rightarrow \{x_1, x_2, \dots, x_p\} \setminus \{x_4\} \Rightarrow \dots$
 - Bidirectional search
- Evaluation metrics :
 - Distances : Euclidean, Manhattan, point scatter matrices, Kullback-Leibler divergence, etc.
 - Information : mutual information, information gain (IG), etc.
 - Correlations : Pearson correlation, Maximal information coefficients (MIC)
- Stopping rules : number of features, number of iterations, non-incremental metrics, attaining optimality, etc.
- Validation and comparison

特征提取：原特征中无，新特征
特征工程 | 特征选择：原特征中有
根据模型和统计数据选择特征
过滤式：先选择特征再训练模型。
封装式：模型训练完后根据好坏动态
选择特征。
嵌入式：特征的选择和学习同时进行。

前向寻找
反向寻找
正反寻找

都是根据每个特征的贡献度
来选择特征。

停止准则

不同特征选择方案的验证和比较。

Three Types of Feature Selection

- Filter : filter the features by their correlations (or MIC, IG) with response variables
- Wrapper : use accuracy, precision, recall, AUC, etc.
 - Akaike Information Criteria (AIC) : $AIC = -2 \ln(L) + 2k$
 - Bayes Information Criteria (BIC) : $BIC = -2 \ln(L) + k \ln(n)$
 - Minimize AIC or BIC, where L is likelihood function, k is the number of features (parameters), n is the number of samples
- Embedded :
 - Random forest : feature importance
 - Regularization : Ridge and LASSO
 - Recursive feature elimination (RFE) : select the best (worst) feature according to the coefficients (e.g. linear regression), then do this recursively to find the feature importance

pearson → 连续 (但是针对的是线性的)

IG → 离散 (可以对非线性的, 但只能离散)

MIC → 连续 (可以非线性)

MIC 是不断分割、密集的 IG. 可以看作是

$$\lim_{\substack{i \rightarrow \infty \\ j \rightarrow \infty}} I_G(i, j)$$

(MIC 在 python 的 minepy 包中)

线性回归有回归系数, 若较大, 敏感度高.