数据降维

机器学习研究室

计算机科学与技术学院
吉林大学

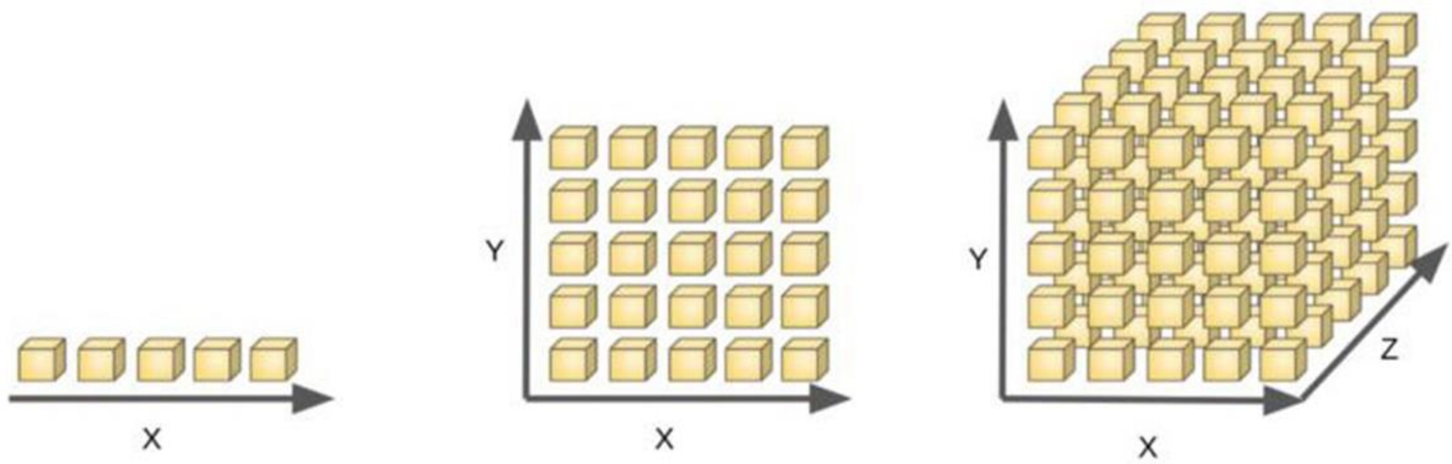
目　　录

- 数据降维概述
- 主成分分析(PCA)算法
- 表征学习(Representation Learning)
- 自编码器

数据降维概述

数据降维概述

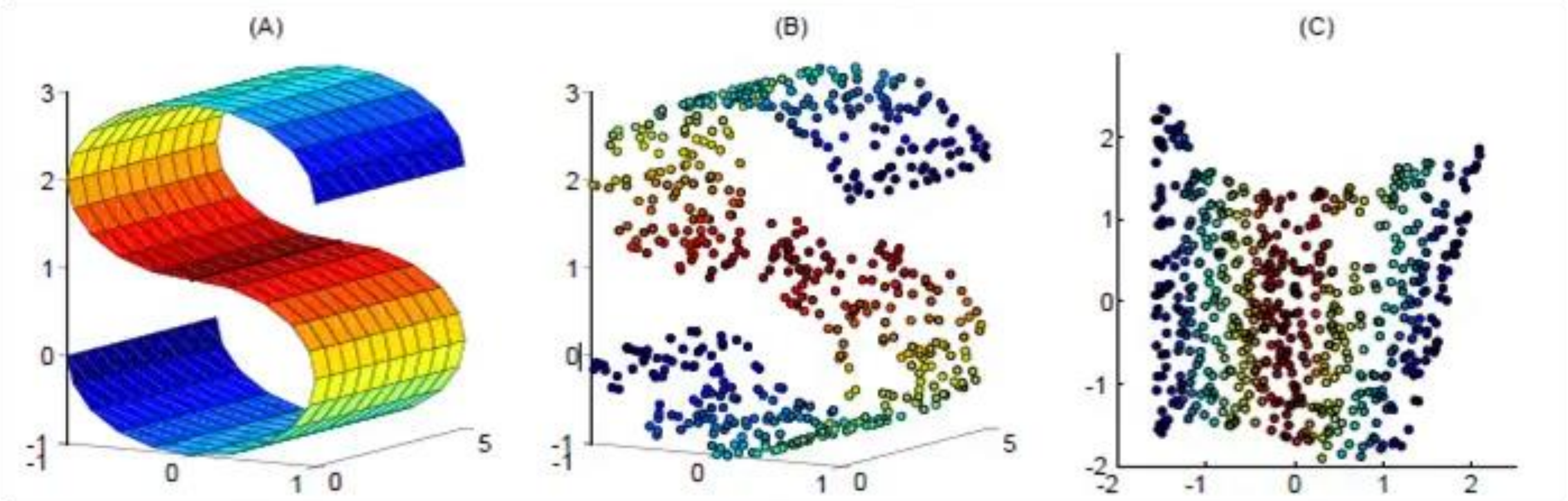
- 维数灾难(Curse of Dimensionality)：通常是指在涉及到向量的计算的问题中，随着维数的增加，计算量呈指数倍增长的一种现象。
- 在很多机器学习问题中，每条数据经常具有很高的特征维度。如果直接使用原始的数据，不仅会让训练非常缓慢，还会影响模型的泛化性能。



数据降维概述

什么是数据降维？

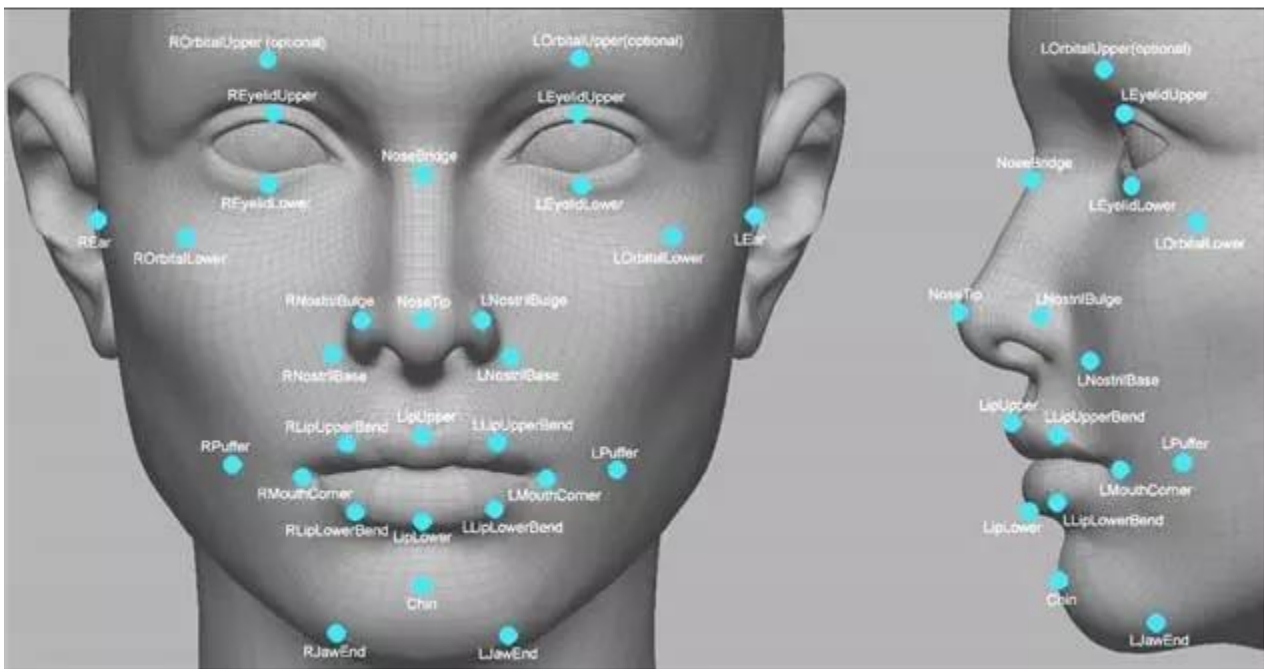
- 降维(Dimensionality Reduction)是将训练数据中的样本(实例)从高维空间转换到低维空间。该过程与信息论中有损压缩概念相似，完全无损的降维是不存在。
- 降维方法又分为线性降维和非线性降维，非线性降维又分为基于核函数和基于流形等方法。



数据降维概述

为什么要降维？

- 数据降维可以使得数据集更易使用、确保变量之间彼此独立、降低算法计算运算成本、去除噪音。
- 数据降维常应用于文本处理、人脸识别、图片识别、自然语言处理等领域。

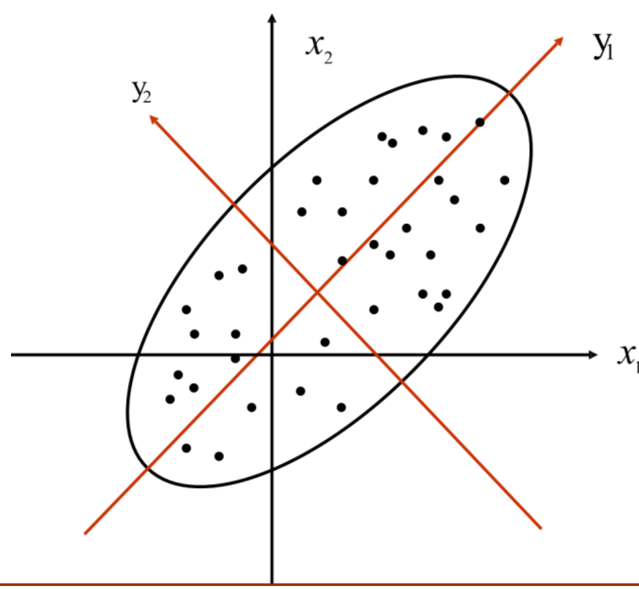


主成分分析(PCA)算法

PCA算法的思想

- 通过平移、旋转坐标轴完成对数据原始特征空间的重构

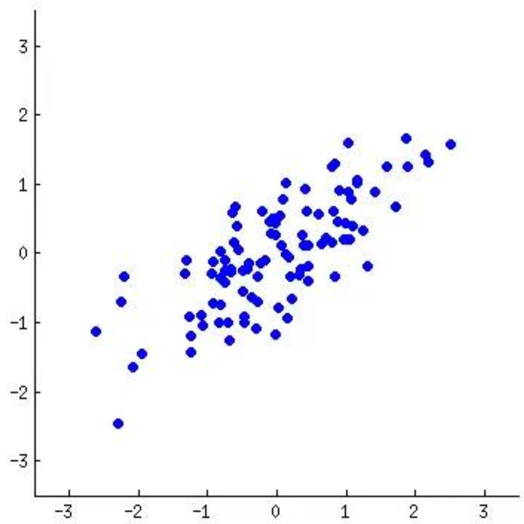
$$
\begin{array}{c|c}
 & \begin{matrix} \text{数据}1 & \text{数据}2 & & \text{数据}m \end{matrix} \\
\begin{matrix} \text{特征}1 \\ \text{特征}2 \\ \vdots \\ \text{特征}n \end{matrix} &
\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}_{n\times m}
\end{array}
\xrightarrow{\text{平移}\oplus\text{旋转}}
\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1m} \\ y_{21} & y_{22} & \cdots & y_{2m} \\ \vdots & \vdots & & \vdots \\ y_{k1} & y_{k2} & \cdots & y_{km} \end{bmatrix}_{k\times m}
$$

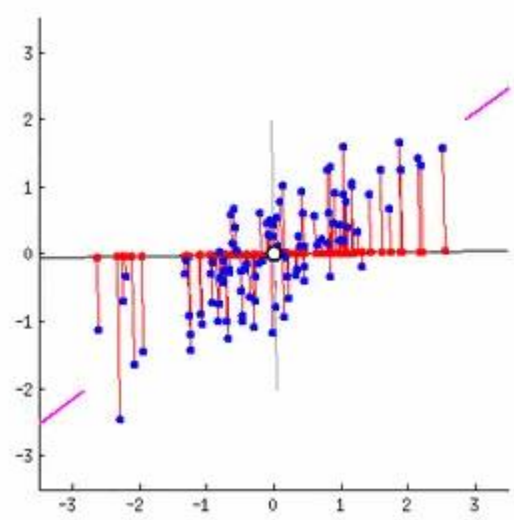


PCA算法的思想

- PCA算法对于重构和降维的要求：

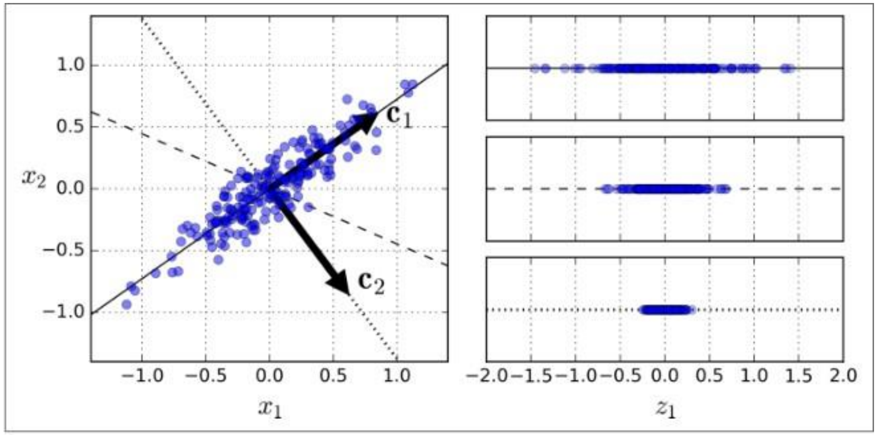
1. 重构的不同维度之间线性无关(正交、协方差为0)；
2. 降维后所得维度的值尽可能分散(最大方差)；





PCA算法的思想

- 一个使用PCA算法的实例
1. 识别在数据集中最大方差量的轴(c1)；
2. 找到与第一个轴正交的第二个轴(c2)。



- 对于高维数据按照规则继续计算；
- 第 $\boldsymbol{i}$ 轴的单位向量称为第 $\boldsymbol{i}$ 个主成分 (PC) ；
- 例子中第一个 PC为c_1，第二个 PC 为c_2 。

PCA算法推导

- 记X是一个有m个样本点和n个特征的数据表

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}_{n\times m}$$

⇩

$$\mathbf{X} = (x_{ij})_{n\times m} = \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_n \end{pmatrix} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_{\boldsymbol{m}})$$

样本点： $\boldsymbol{e}_j = (x_{1j}, x_{2j}, \cdots, x_{nj})^T \in \mathbf{R}^n \ (j = 1,2,\cdots,m)$

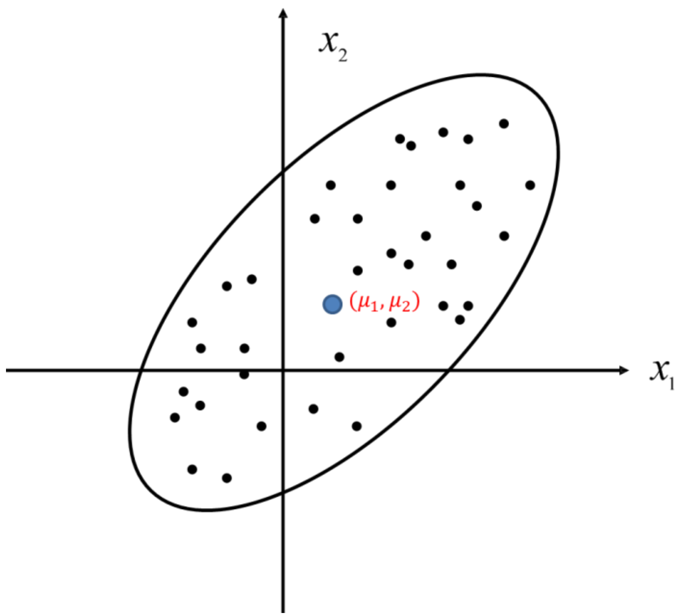
特征： $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \cdots, x_{im}) \in \mathbf{R}^m \ (i = 1,2,\cdots,n)$

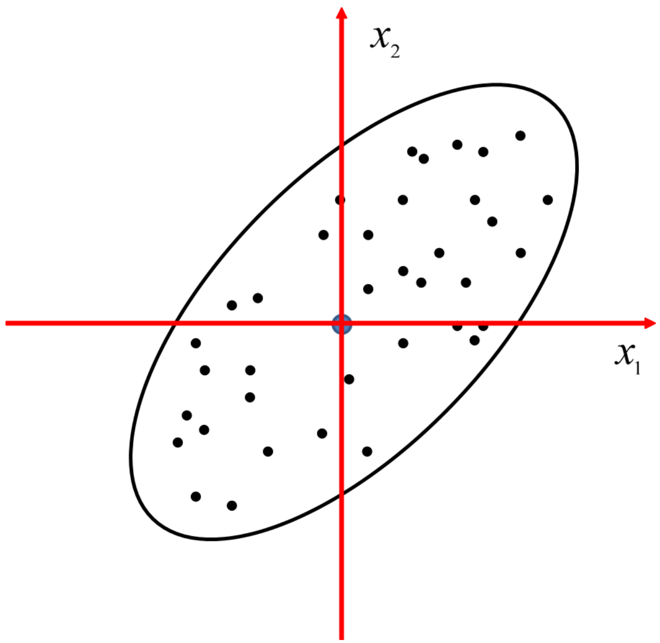
PCA算法推导

- 平移操作(中心化处理)：将每个变量的均值都化为 0
- 对于任意特征 $\boldsymbol{x}_i$，进行如下变换：

$$\boldsymbol{x}_i = \boldsymbol{x}_i - \mu_i$$

其中μ_j为所有数据点在特征j上的均值。



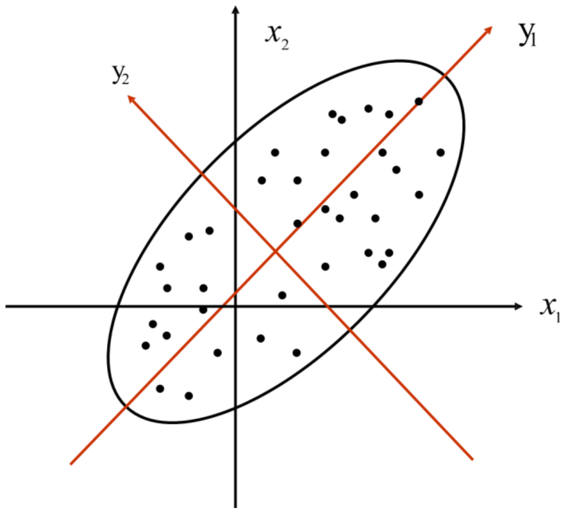


PCA算法推导

- 旋转操作(坐标变换)：
- 将矩阵X中的每一列向量变换到矩阵P中以每一行行向量为基所表示的空间中去。

$$\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1m} \\ y_{21} & y_{22} & \cdots & y_{2m} \\ \vdots & \vdots & & \vdots \\ y_{k1} & y_{k2} & \cdots & y_{km} \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & & \vdots \\ p_{k1} & p_{k2} & \cdots & p_{kn} \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}$$

Y P X



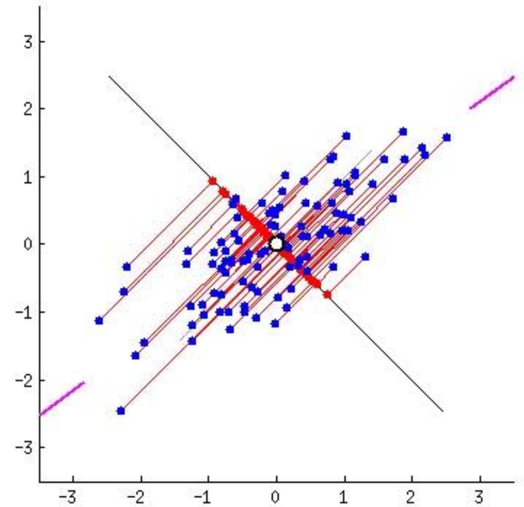
PCA算法推导

- 降维后所得维度的值尽可能分散(方差最大化)
- 任意特征y_i的方差计算公式：

$$\mathrm{Var}(y_i)=\frac{1}{m-1}\sum_{j=1}^{m}\left(y_{ij}-\mu_i\right)^2$$

- 由于之前进行中心化处理，因此公式可简写为：

$$\mathrm{Var}(y_i)=\frac{1}{m-1}\sum_{j=1}^{m}{y_{ij}}^2$$



PCA算法推导

- 重构的不同维度之间线性无关(协方差为0)
- 任意特征y_i, y_j的协方差计算公式：

$$\mathrm{Cov}(y_i, y_j) = \frac{1}{m-1}\sum_{k=1}^{m} y_{ik} \cdot y_{jk}$$

- 变换之后的协方差矩阵：

$$D = \frac{1}{m-1}YY^T = \begin{bmatrix} \mathrm{Var}(y_1) & \mathrm{Cov}(y_1, y_2) & \cdots & \mathrm{Cov}(y_1, y_k) \\ \mathrm{Cov}(y_2, y_1) & \mathrm{Var}(y_2) & \cdots & \mathrm{Cov}(y_2, y_k) \\ \vdots & \vdots & & \vdots \\ \mathrm{Cov}(y_k, y_1) & \mathrm{Cov}(y_k, y_2) & \cdots & \mathrm{Var}(y_k) \end{bmatrix}$$

- 目标为找到一种变换P，使得变换之后的数据Y的协方差矩阵D满足：除对角线外的其它元素化为 0，并且在对角线上将元素按大小从上到下排列（变量方差尽可能大）

PCA算法推导

- 如何找到变换P，使协方差矩阵为对角矩阵？

利用变换前X的协方差矩阵C和变换之后Y的协方差矩阵D的关系

$$
\begin{aligned}
\mathrm{D} &= \frac{1}{m-1} Y Y^T \\
&= \frac{1}{m-1} (PX)(PX)^T \\
&= \frac{1}{m-1} P X X^T P^T \\
&= P\left(\frac{1}{m-1} X X^T\right) P^T \\
&= P C P^T
\end{aligned}
$$

- 优化目标变成了：寻找一个矩阵 P, 满足PCP^T是一个对角矩阵, 并且对角元素按从大到小依次排列, 那么P的前k行就是要寻找的基变换, 用P的前k行组成的矩阵乘以X就使得X从n维降到了k维并满足上述优化条件。

PCA算法推导

- 特征值分解求解（线性代数）
- 协方差矩阵C是一个是n行n列的实对称矩阵，因此可以找到n个单位正交特征向量, 设这n个特征向量为e_1, e_2 , $\cdots, e_n$我们将其按列组成矩阵:

$$P = (e_1 \quad e_2 \quad \cdots \quad e_n)$$

则协方差矩阵C可以被P对角化:

$$P^T CP = \Lambda = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix}$$

其对角元素为各特征向量对应的特征值。

PCA算法步骤

- 算法步骤：

Algorithm 1 PCA 算法

输入: 数据矩阵 $X=[x_1,x_2\cdots,x_m]\in\mathbb{R}^{n\times m}$, 降维后样本维数 k.

输出: 转换矩阵 $\boldsymbol{P}=(\boldsymbol{p}_1,\boldsymbol{p}_2,\ldots,\boldsymbol{p}_k)^T\in\mathbb{R}^{k\times n}$.

1: **for** $i=1:m$ **do**
2: 　中心化处理样本: $\boldsymbol{x}_i\leftarrow\boldsymbol{x}_i-\boldsymbol{\mu_i}$, 其中 $\boldsymbol{\mu_i}=\frac{1}{n}\sum_{j=1}^{n}\boldsymbol{x}_j$ 为样本均值.
3: **end for**
4: 计算协方差矩阵 $\Sigma=\frac{1}{m-1}\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$.
5: 对协方差矩阵 Σ 做特征值分解 $\Lambda=P\Sigma P^T$ 并将特征值降序排序: $\lambda_1\geqslant\lambda_2\geqslant\ldots\geqslant\lambda_n$.
6: 取最大的前 k 个特征值对应的特征向量 $\boldsymbol{p}_1,\boldsymbol{p}_2,\ldots,\boldsymbol{p}_k$ 组成转换矩阵 $\boldsymbol{P}$.
7: **输出:** 变换矩阵 P.

- 用转换矩阵对X进行变换即可得到降维的数据:

$$Y=PX$$

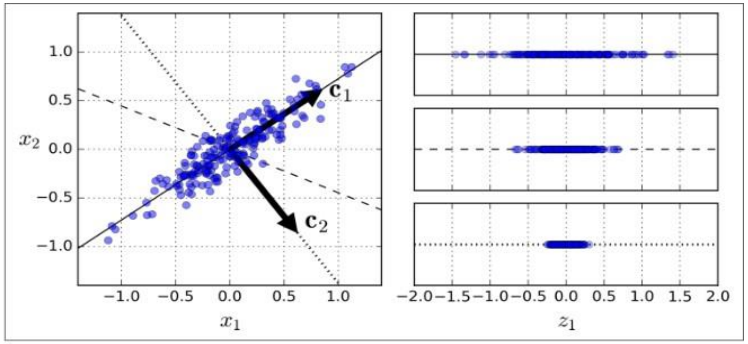
其中$Y\in\mathbb{R}^{k\times m}$。

PCA算法使用

- 主成分
- 降维之后的数据Y为：

$$Y = PX = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1m} \\ y_{21} & y_{22} & \cdots & y_{2m} \\ \vdots & \vdots & & \vdots \\ y_{k1} & y_{k2} & \cdots & y_{km} \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_1 \\ \boldsymbol{y}_2 \\ \vdots \\ \boldsymbol{y}_k \end{bmatrix}$$

- 主成分的定义：变换后的数据，方差最大的变量$\boldsymbol{y}_1$为原始数据X的第一主成分，$\boldsymbol{y}_k$为原始数据X的第k主成分。



PCA算法使用

- 方差贡献率
- 第k主成分$\boldsymbol{y}_k$的方差贡献率定义为$\boldsymbol{y}_k$的方差与所有方差之和的比，记作：

$$\eta_k = \frac{\lambda_k}{\sum_{i=1}^{n} \lambda_i}$$

- 前k个主成分$\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_k$的累计方差贡献率定义为$k$个方差之和与所有方差之和的比：

$$\sum_{i=1}^{k} \eta_i = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i}$$

- 通常取k使得累计方差贡献率达到规定的百分比以上，如 70%~80%以上。

PCA算法使用

- PCA的算法案例

$$X = \begin{pmatrix} -1 & -1 & 0 & 2 & 0 \\ -2 & 0 & 0 & 1 & 1 \end{pmatrix}$$

- 以这个为例，我们用PCA的方法将这组二维数据降到一维

- 因为这个矩阵的每行已经是零均值，所以我们可以直接求协方差矩阵：

$$C = \frac{1}{5}\begin{pmatrix} -1 & -1 & 0 & 2 & 0 \\ -2 & 0 & 0 & 1 & 1 \end{pmatrix}\begin{pmatrix} -1 & -2 \\ -1 & 0 \\ 0 & 0 \\ 2 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{6}{5} & \frac{4}{5} \\ \frac{4}{5} & \frac{6}{5} \end{pmatrix}$$

PCA算法使用

- PCA的算法案例
- 求C的特征值和特征向量：

$$|C-\lambda I| = \begin{vmatrix} \frac{6}{5}-\lambda & \frac{4}{5} \\ \frac{4}{5} & \frac{6}{5}-\lambda \end{vmatrix} = (\frac{6}{5}-\lambda)^2 - \frac{16}{25} = (\lambda-2)(\lambda-2/5) = 0$$

- 求解得到特征值：

$$\lambda_1 = 2，\lambda_2 = 2/5$$

- 其对应的特征向量分别是：

$$P_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}，P_2 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

PCA算法使用

- PCA的算法案例

由于对应的特征向量分别是一个通解， P_1和P_2可取任意实数。那么标准化后的特征向量为：

$$\begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}, \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$$

因此我们的矩阵P是：

$$P = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix}$$

PCA算法使用

- PCA的算法案例
- 可以验证协方差矩阵P的对角化：

$$PCP^{\mathrm{T}}=\begin{pmatrix}1/\sqrt{2} & 1/\sqrt{2}\\ -1/\sqrt{2} & 1/\sqrt{2}\end{pmatrix}\begin{pmatrix}6/5 & 4/5\\ 4/5 & 6/5\end{pmatrix}\begin{pmatrix}1/\sqrt{2} & -1/\sqrt{2}\\ 1/\sqrt{2} & 1/\sqrt{2}\end{pmatrix}$$

$$=\begin{pmatrix}2 & 0\\ 0 & 2/5\end{pmatrix}$$

- 最后我们用P的第一行乘以数据矩阵，就得到了降维后的数据表示：

$$Y=\begin{pmatrix}1/\sqrt{2} & 1/\sqrt{2}\end{pmatrix}\begin{pmatrix}-1 & -1 & 0 & 2 & 0\\ -2 & 0 & 0 & 1 & 1\end{pmatrix}$$

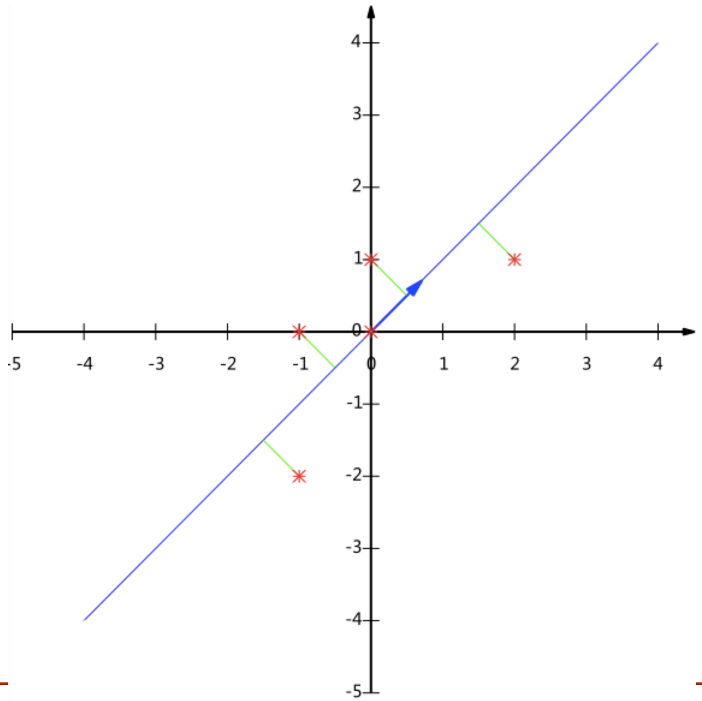
$$=\begin{pmatrix}-3/\sqrt{2} & -1/\sqrt{2} & 0 & 3/\sqrt{2} & 1/\sqrt{2}\end{pmatrix}$$

PCA算法使用

- PCA的算法案例
- 降维后的投影结果如下图：

$$Y = (1/\sqrt{2} \quad 1/\sqrt{2}) \begin{pmatrix} -1 & -1 & 0 & 2 & 0 \\ -2 & 0 & 0 & 1 & 1 \end{pmatrix}$$

$$= (-3/\sqrt{2} \quad -1/\sqrt{2} \quad 0 \quad 3/\sqrt{2} \quad 1/\sqrt{2})$$



PCA算法的优缺点

- PCA算法优点

1. 仅仅需要以方差衡量信息量,不受数据集以外的因素影响
2. 各主成分之间正交,可消除原始数据成分间的相互影响的因素
3. 计算方法简单,主要运算时特征值分解,易于实现
4. 它是无监督学习,完全无参数限制的

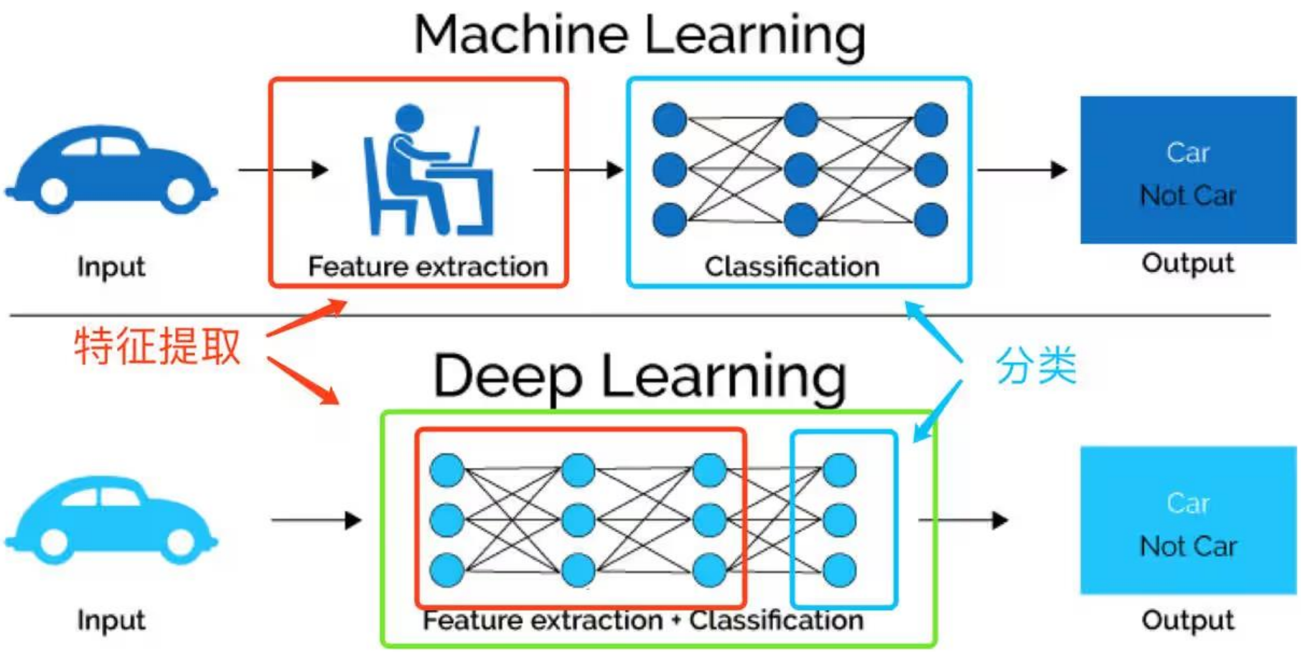
- PCA算法缺点

1. 主成分各个特征维度的含义具有一定的模糊性,不如原始样本特征的解释性强
2. 方差小的非主成分也可能含有对样本差异的重要信息,因降维丢弃可能对后续数据处理有影响

表征学习

表征学习

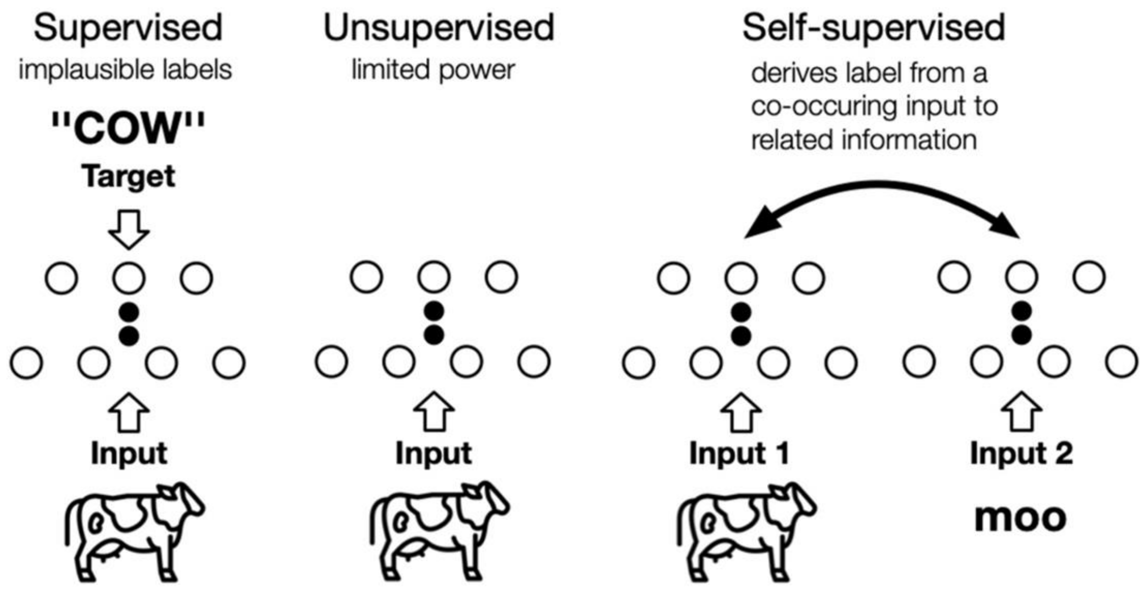
- 表征学习(Representation Learning)，也称为特征学习，是机器学习中的一种技术，其目标是从原始数据中自动发现特定任务所需的表征或特征。



表征学习

表征学习的类型：

- 监督表征学习：使用标记数据来学习表征。
- 无监督表征学习：使用未标记的数据来学习表征。
- 自监督表征学习：构造监督信息来学习表征。



表示学习：可以从未标记的数据集学习良好的数据表示。

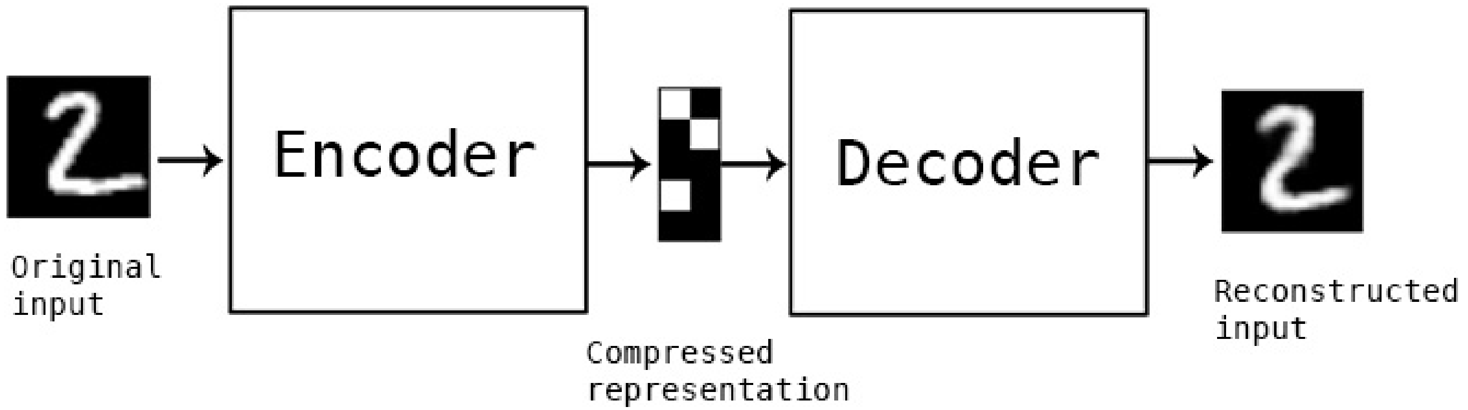
表征学习

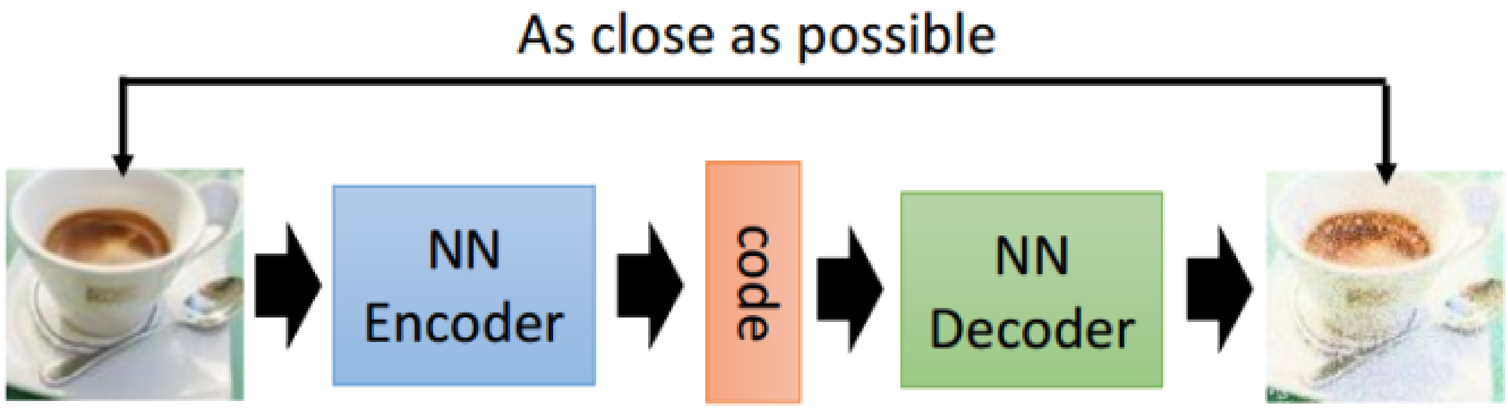
自监督表征学习：

- 基于生成的自监督学习
  - ➢自编码器(AutoEncoder)
  - ➢变分自编码器(VAE)
  - ➢生成对抗网络(GAN)
- 基于自预测的自监督学习
  - ➢基于Mask的自预测
  - ➢基于多视图间的自预测
  - ➢基于固有关系的自预测
- 基于对比学习的自监督学习
  - ➢直接对比：PIRL, SimCLR, MoCo
  - ➢基于聚类：DeepCluster、SeLA、SwAV
  - ➢基于知识蒸馏：BYOL, SimSiam

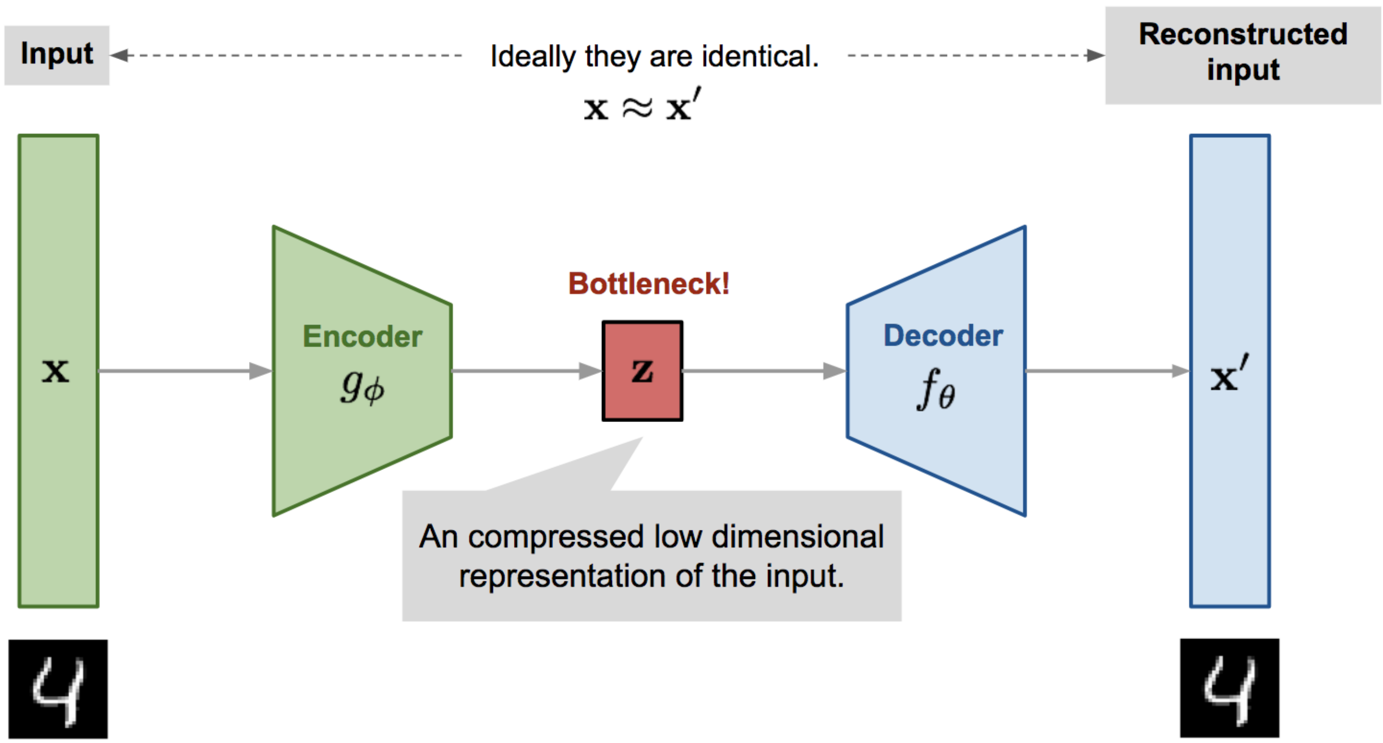
AUTOENCODER(自编码器)

AutoEncoder(自编码器)





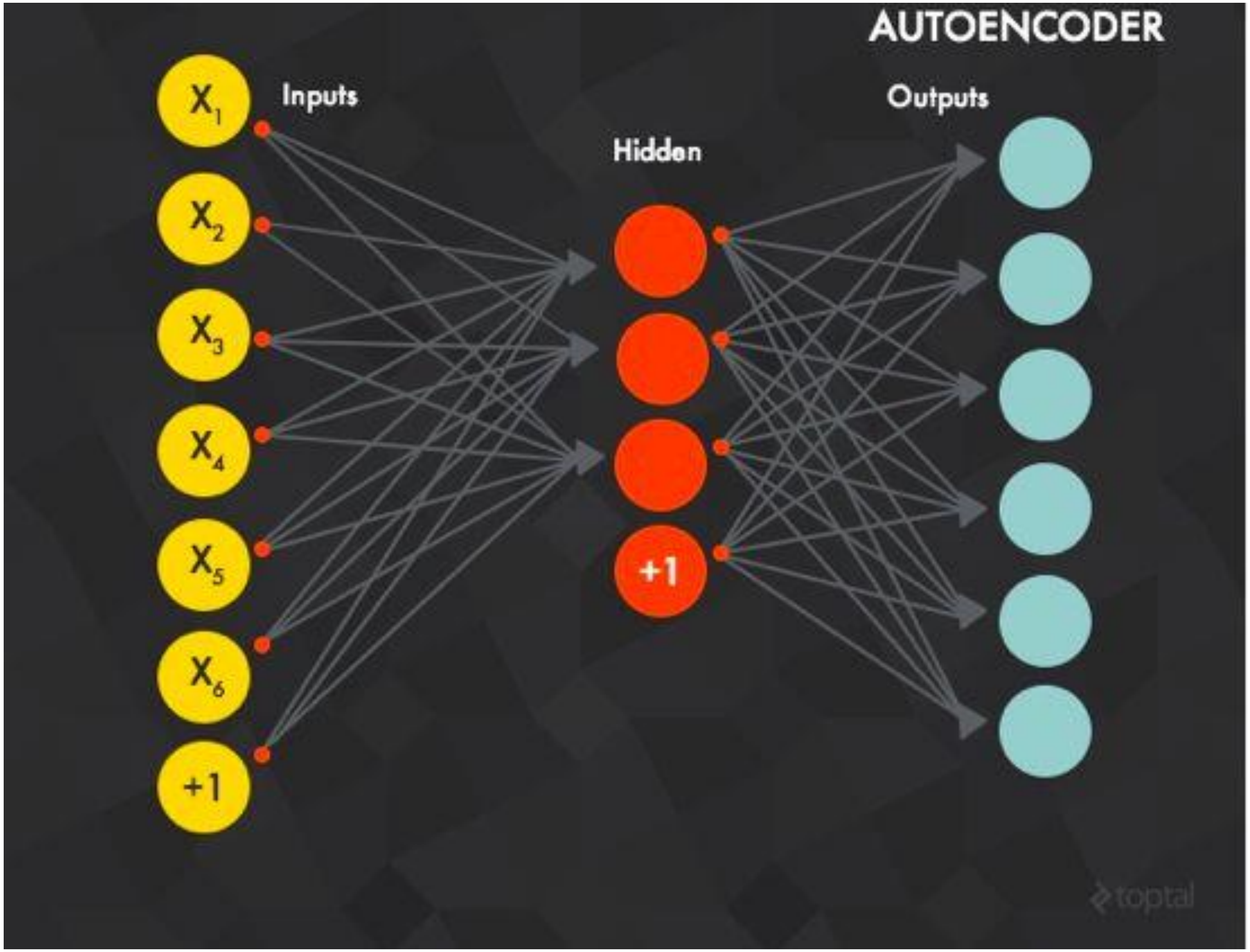
AutoEncoder(自编码器)



$$L_{\text{AE}}(\theta, \phi) = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}^{(i)} - f_\theta(g_\phi(\mathbf{x}^{(i)})))^2$$

Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks[J]. science, 2006, 313(5786): 504-507.

AutoEncoder(自编码器)



AutoEncoder实例

- MNIST数据集

http://yann.lecun.com/exdb/mnist/

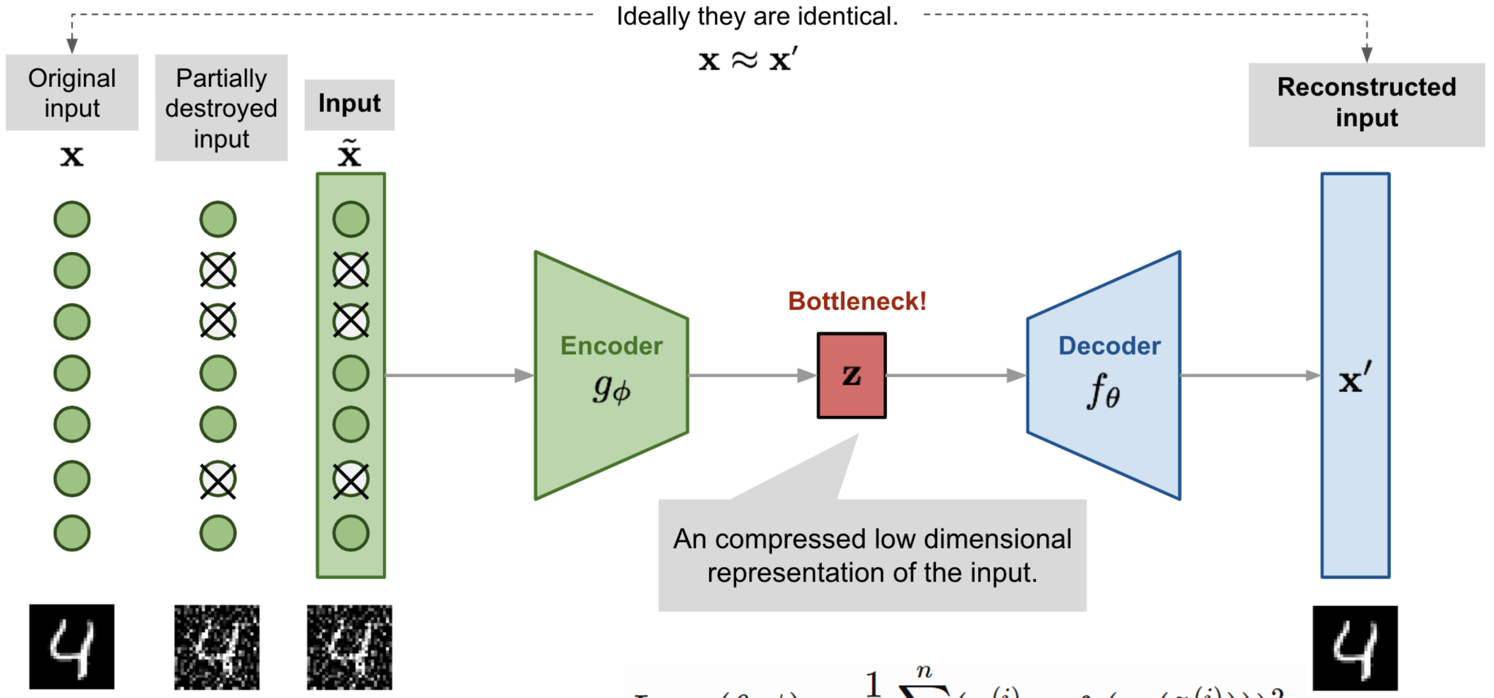
AutoEncoder实例

Epoch 10

Epoch 90

Denoising AutoEncoder

AutoEncoder存在过拟合的风险
解决办法：输入中加入随机噪声

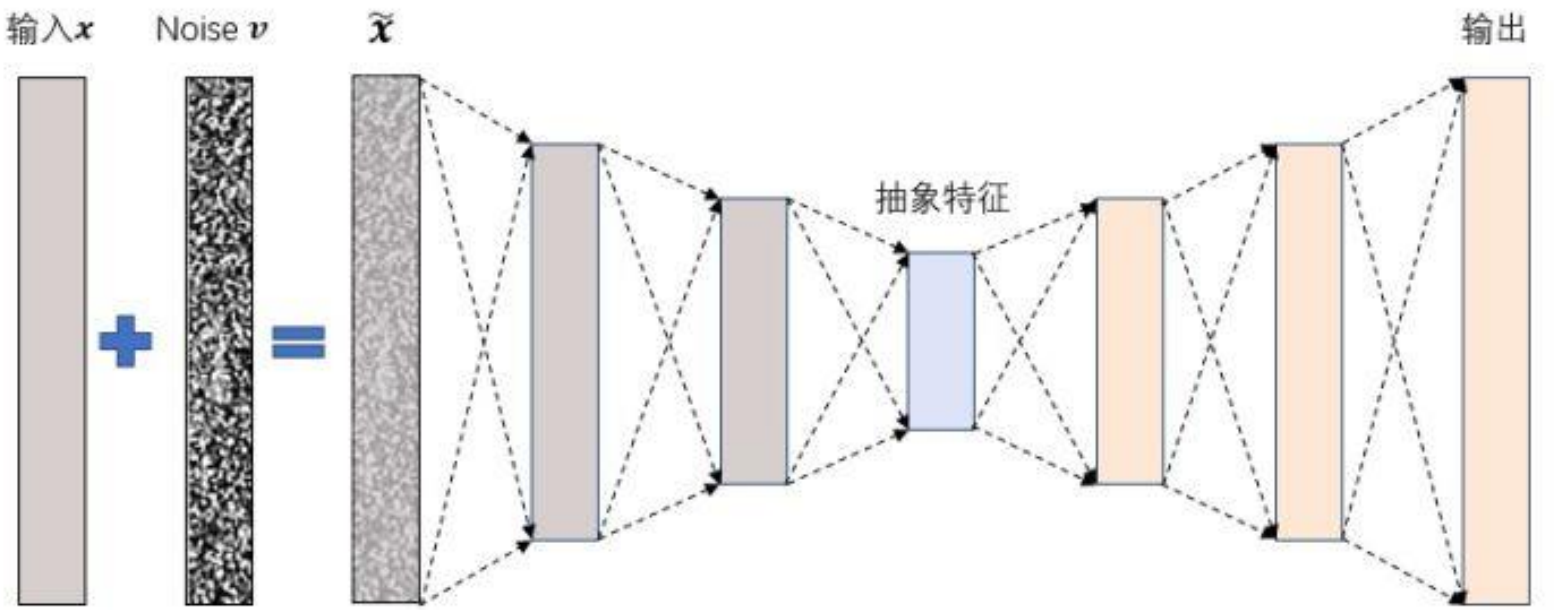


$$L_{\mathrm{DAE}}(\theta, \phi) = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}^{(i)} - f_\theta(g_\phi(\tilde{\mathbf{x}}^{(i)})))^2$$

Vincent P, Larochelle H, Bengio Y, et al. Extracting and composing robust features with denoising autoencoders[C]//Proceedings of the 25th international conference on Machine learning. ACM, 2008: 1096-1103.

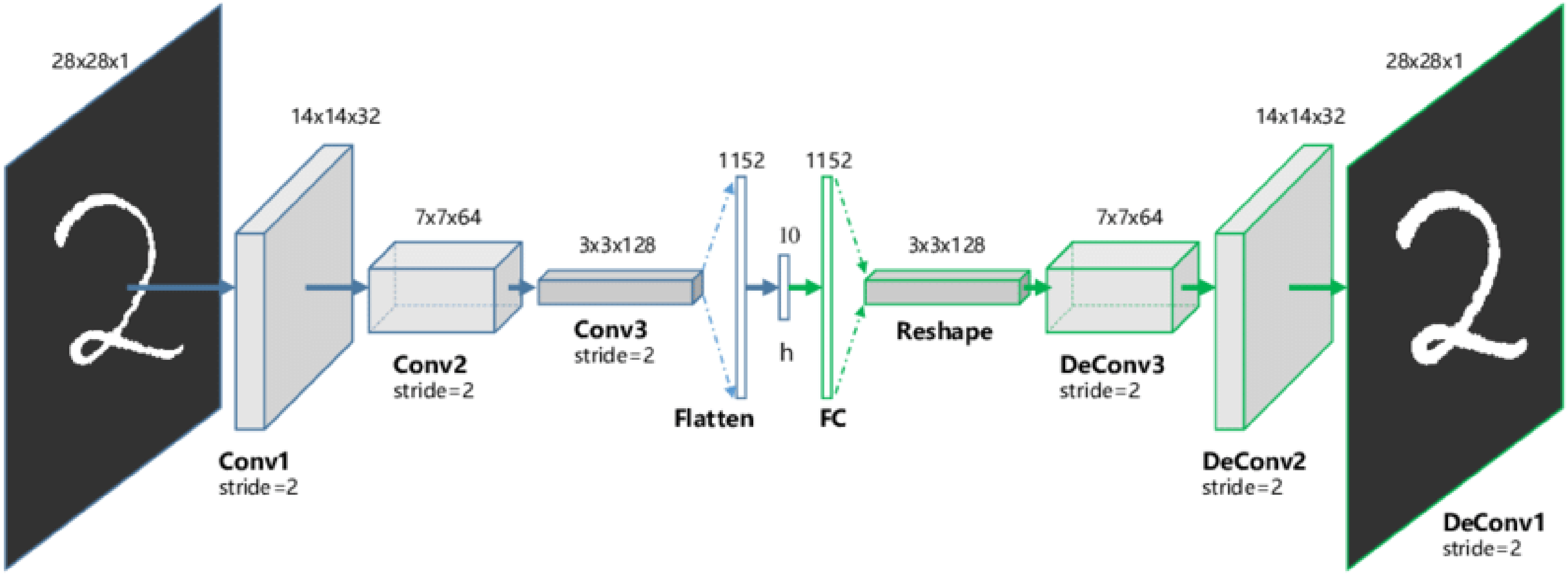
Denoising AutoEncoder

AutoEncoder存在过拟合的风险
解决办法：输入中加入随机噪声



Vincent P, Larochelle H, Bengio Y, et al. Extracting and composing robust features with denoising autoencoders[C]//Proceedings of the 25th international conference on Machine learning. ACM, 2008: 1096-1103.

Convolutional Autoencoder



Masci J, Meier U, Cireşan D, et al. Stacked convolutional auto-encoders for hierarchical feature extraction[C]//International Conference on Artificial Neural Networks. Springer, Berlin, Heidelberg, 2011: 52-59.

Convolutional Autoencoder

- Convolutional Autoencoder实例

Epoch 10

Epoch 90

谢谢