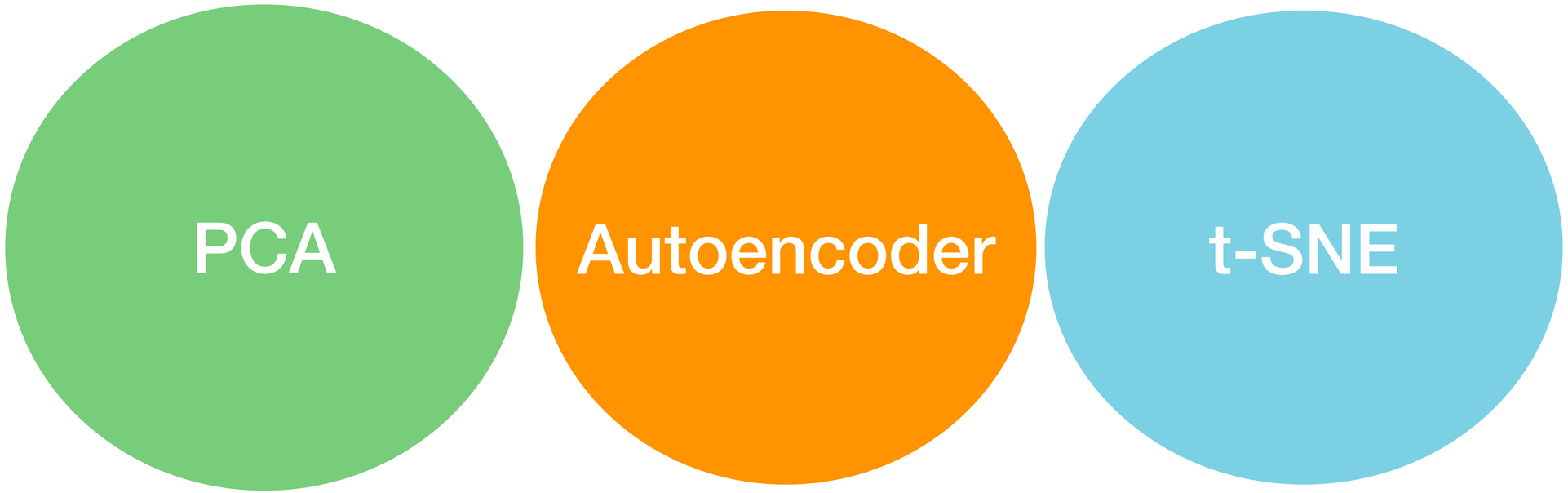
Dimension Reduction

PCA、t-SNE、Autoencoder

Outline



如果要預測

顧客信用卡消費明細

> 15,000 間店家

	國泰人壽	福容大飯店	台塑加油站	大潤發	高鐵	HOLA	…
顧客 1	1	0	0	0	0	0	
顧客 2	0	0	0	0	1	1	
顧客 3	0	0	1	0	0	0	…
…	…	…	…	…	…	…	
顧客 n	0	1	0	1	0	1	

> 640,000 位顧客

預測變數

- 男性 / 女性
- 公教人員
- 薪轉戶
- 理財 VIP 會員
- 世界卡顧客
- 行員
- …

n 個 feature 轉換為 n 個 component

但全部都用的到嗎？

如果資料量更大、記憶體更小呢？

能不能只用部分的特徵數，去

完美詮釋所有的資料，這就是降維

那什麼是 PCA？
(Principal Component Analysis)

將具有N個特徵空間的樣本，轉換為具有K個特徵空間的樣本，其中K < N(通常K=2)，並找到一個特徵空間，使得某點與投影在特徵空間上的點距離最小

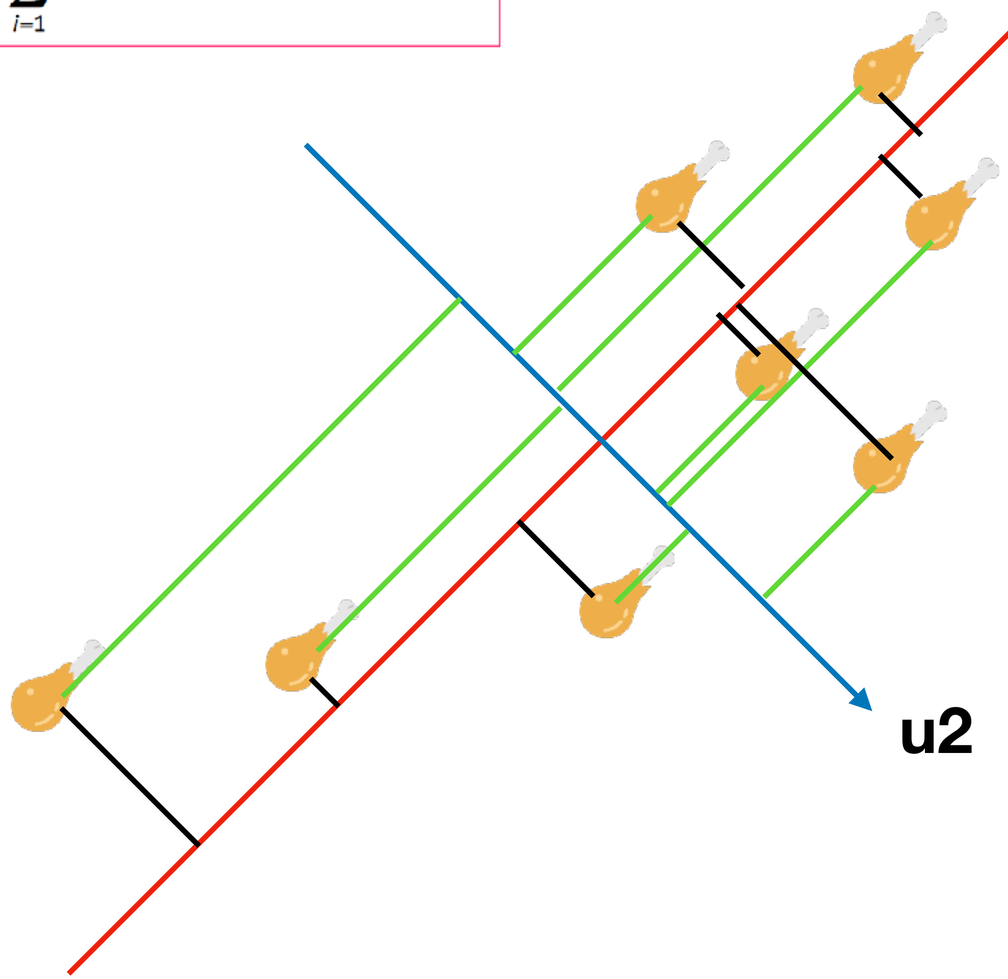

聽不懂 沒關係
林俊傑
OFFICIAL
Audio
Taihe Music

把一個漢堡完全壓扁，在這個壓扁的平面上以剩下的兩軸(主成份)，重建一個平面座標系來表達雞腿的位置

PCA: given M<d. Find $(u_1..u_M)$

that minimizes $E_M \equiv \sum_{k=1}^{d} \left\| x_k - \hat{x}_k \right\|_2^2$

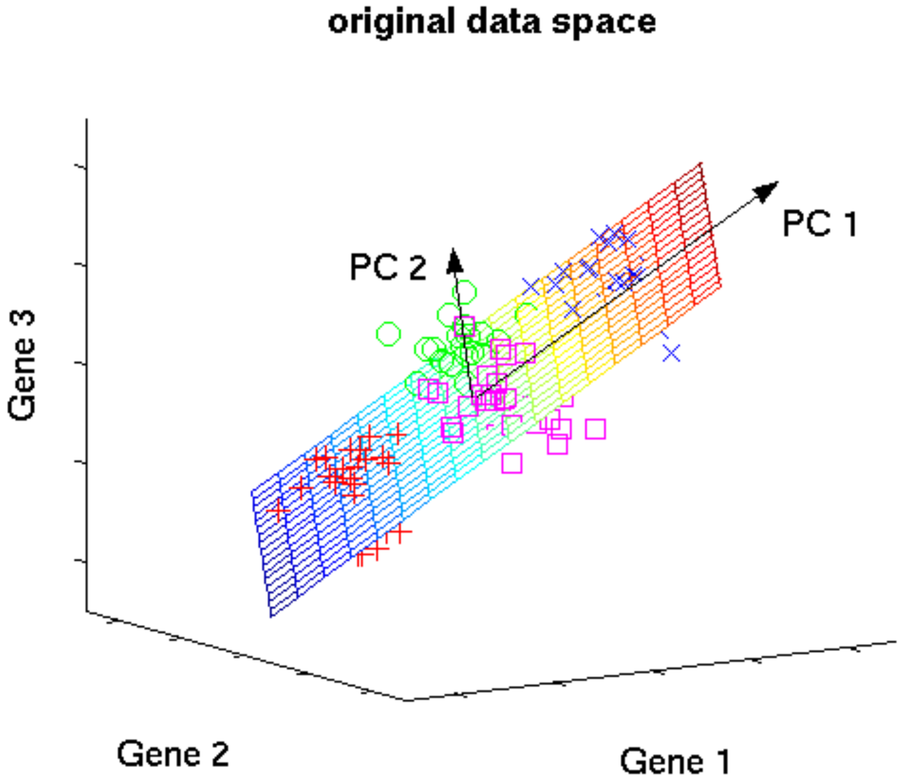
where $\hat{x}_k = \bar{x} + \sum_{i=1}^{M} z_i^k u_i$



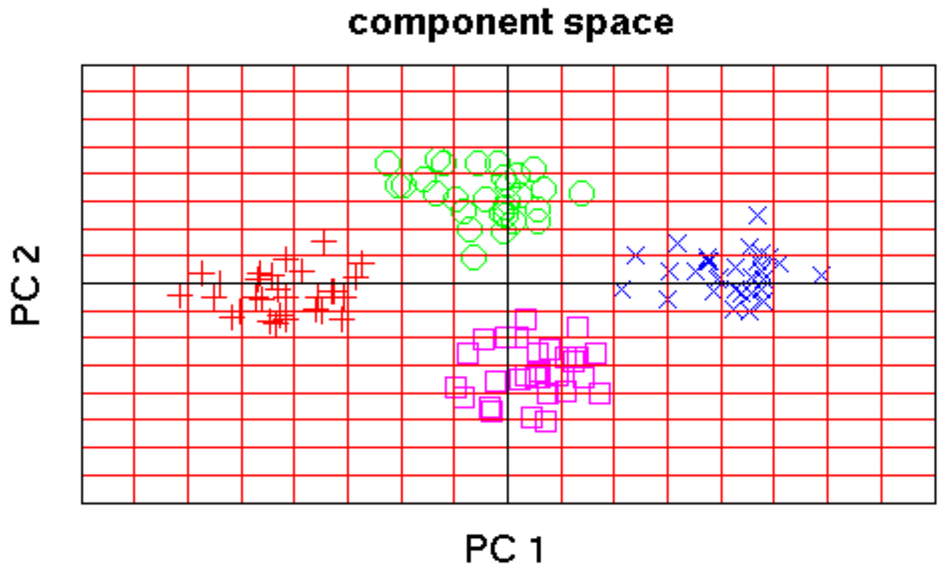
u1 作為主成份好過 u2

PCA

通常取前2名主成份作降維代表



PCA



透過PCA

顧客信用卡消費明細

> 15,000 間店家

> 640,000 位顧客

	國泰人壽	福容大飯店	…
顧客 1	1	0	…
顧客 2	0	0	
顧客 3	0	0	
…	…	…	
顧客 n	0	1	

主成分分析 (降維)

Components

顧客人數

	Componet 1	Componet 2	…	Componet k
顧客 1	0.3	0.0	…	-0.5
顧客 2	-0.4	0.4	…	-0.2
顧客 3	-0.3	0.8	…	0.7
…	…	…	…	…
顧客 n	0.1	-0.2	…	0.3

預測模型

使用 Logistic Regression

$$\log \frac{p}{1\ p} = + {}_1C_1 + \ldots + {}_kC_k$$

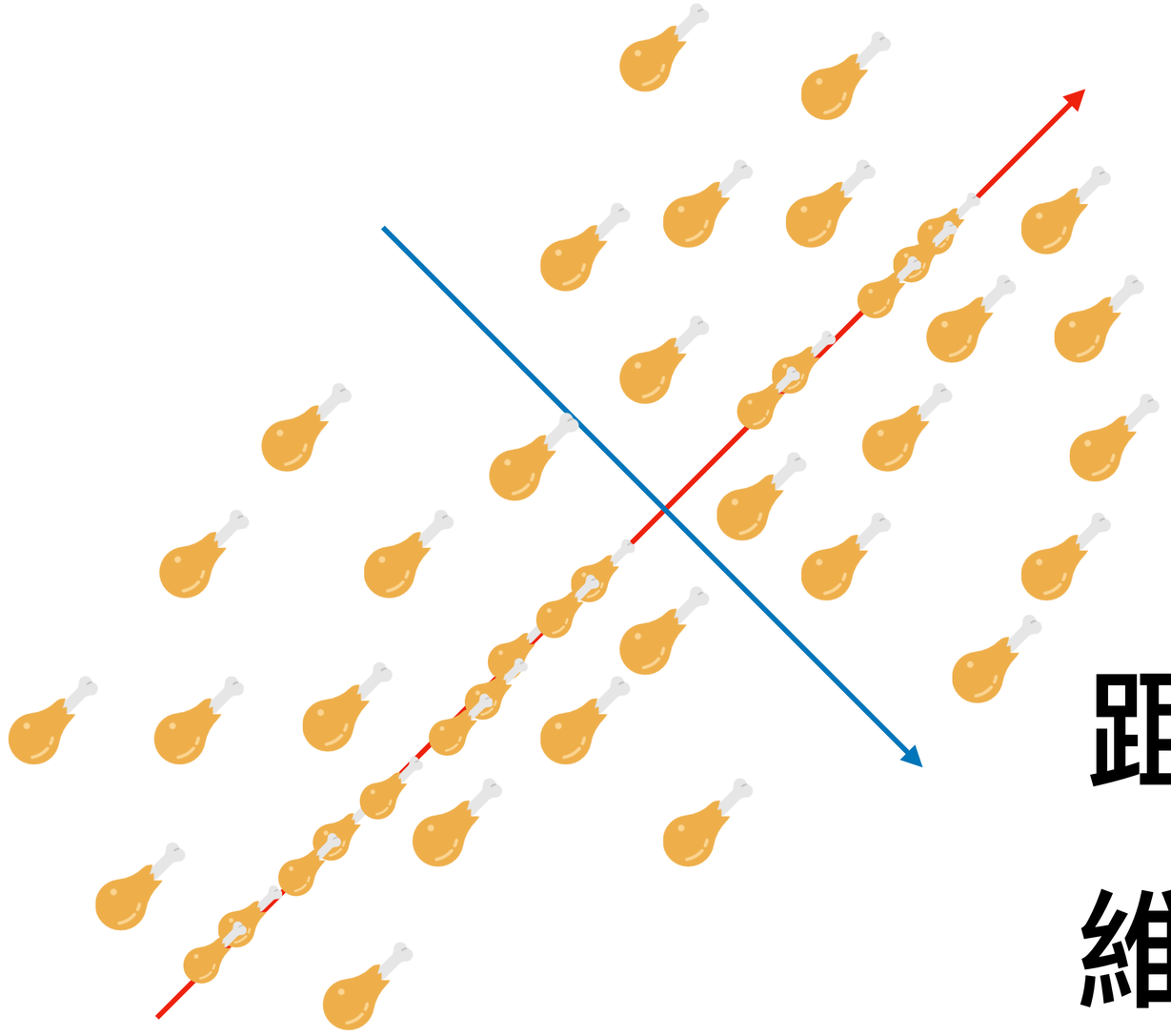
預測變數

男性 / 女性、是否為公教人員、薪轉戶、理財 VIP 會員、世界卡顧客、行員

PCA的限制

是一種線性降維的方式，若特徵間為非線性關係，容易造成Underfitting

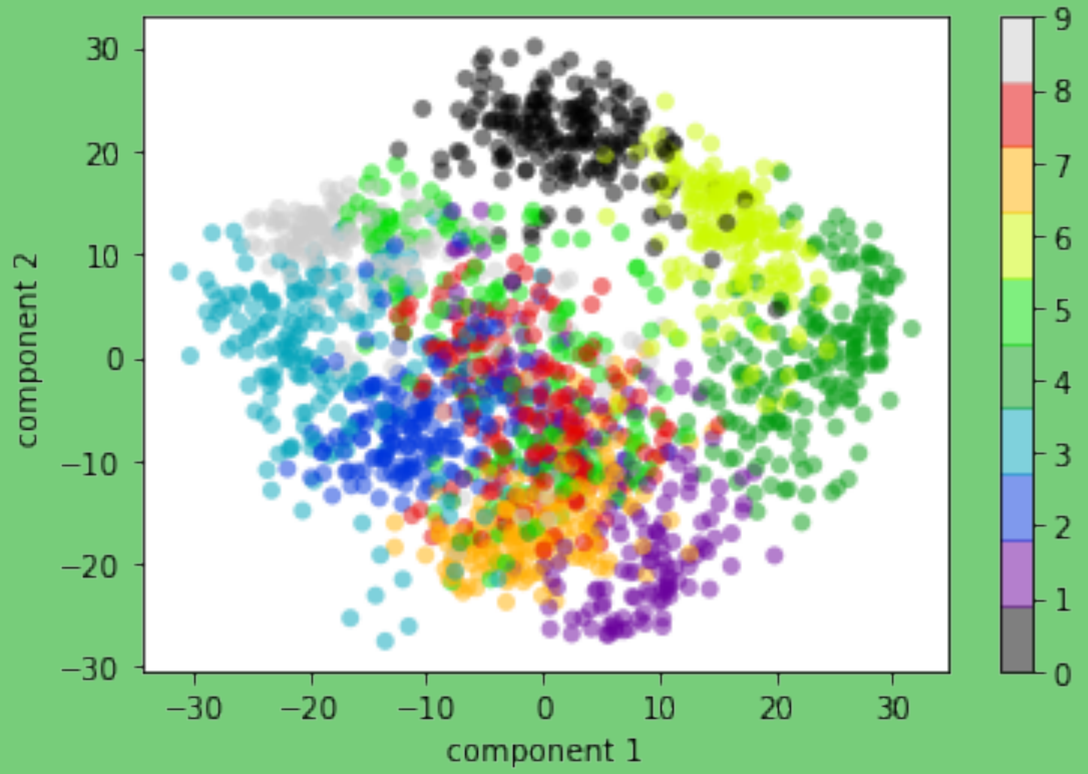


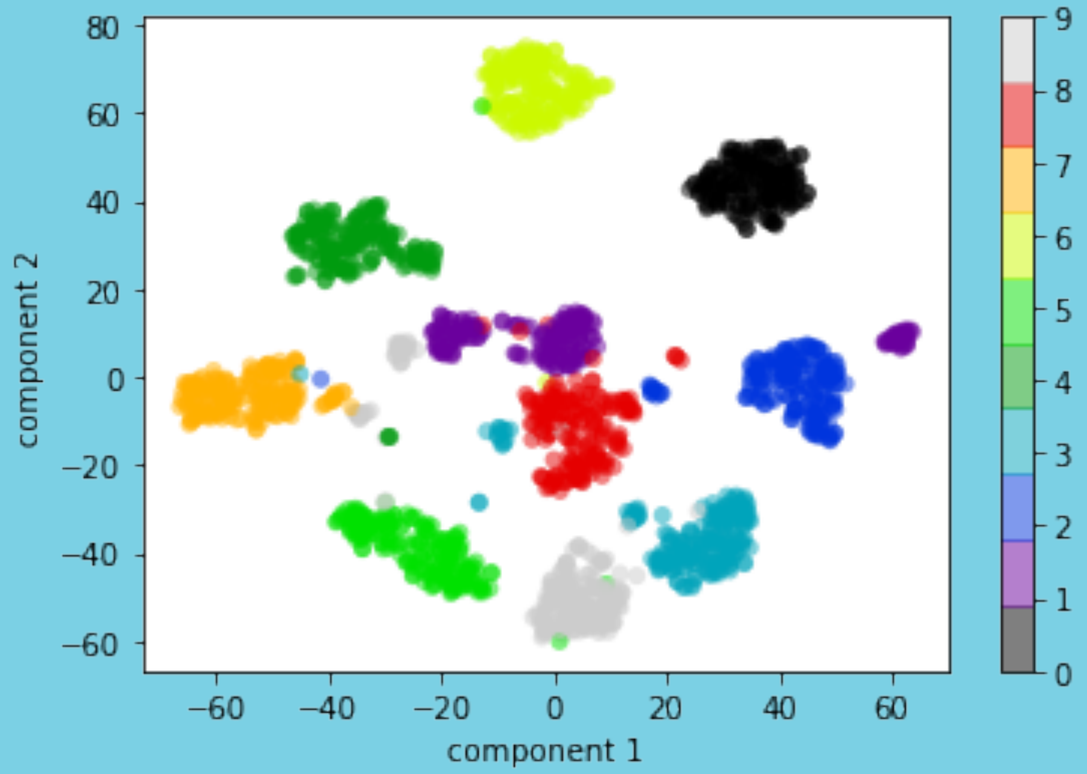
距離近的雞腿，降到低維會重疊**無法辨識**，

有沒有其他映射方式

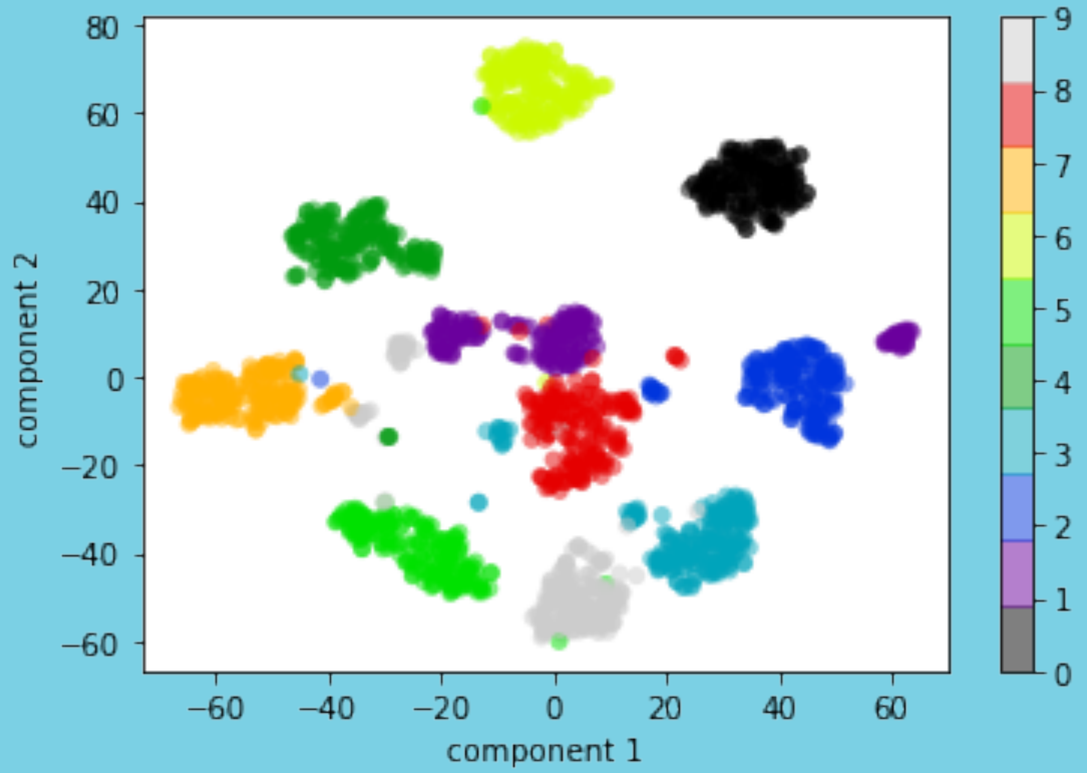
PCA


component 2
component 1
30
20
10
0
−10
−20
−30
−30
−20
−10
0
10
20
30
9
8
7
6
5
4
3
2
1
0

t-SNE


component 2
component 1
80
60
40
20
0
−20
−40
−60
−60
−40
−20
0
20
40
60
9
8
7
6
5
4
3
2
1
0

t-SNE


80
60
40
20
0
−20
−40
−60
component 2
−60
−40
−20
0
20
40
60
component 1
9
8
7
6
5
4
3
2
1
0

那什麼是 t-SNE？
(t-distributed stochastic neighbor embedding)

在講t-SNE之前，

請容許我先介紹SNE

以往表示相似性的做法，都是用歐式距離，而SNE的概念是，把這種距離關係轉換為以條件機率來表示。在高維空間相似的數據點，映射到低維空間距離也是相似的。

高維

Xi 

Xj 

 越靠近  ，Pj|i 越大

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|x_i - x_k\|^2/2\sigma_i^2)}$$

$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k\neq i}\exp(-\|y_i - y_k\|^2)}$$

低維

Yi 

Yj 

 越靠近  ，Qj|i 越大

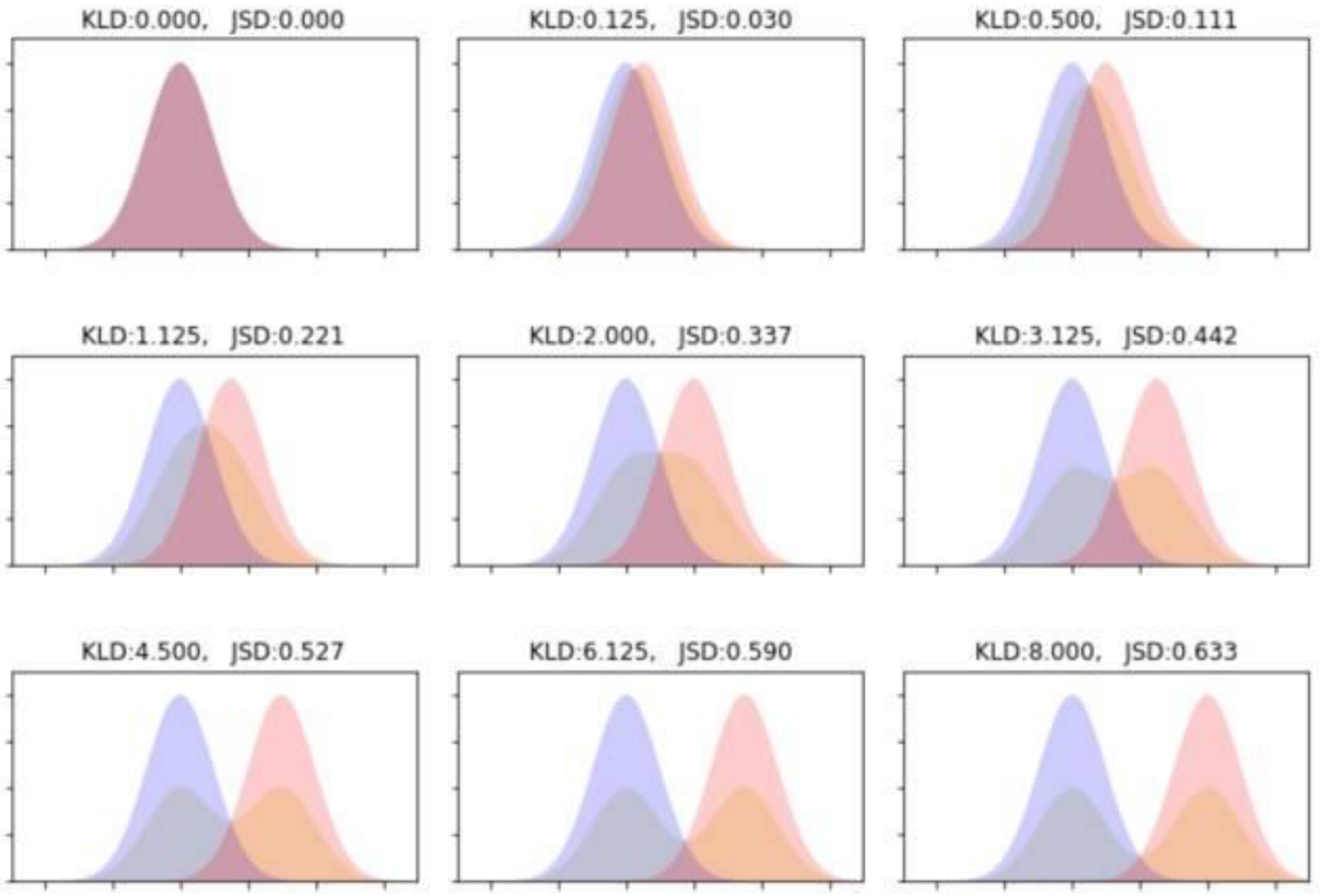
目標：Pj|i、Qj|i 分佈越相近越好

怎麼做：KL散度

KL散度

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

KL散度越大，分布相似度越低



問題1

越靠近

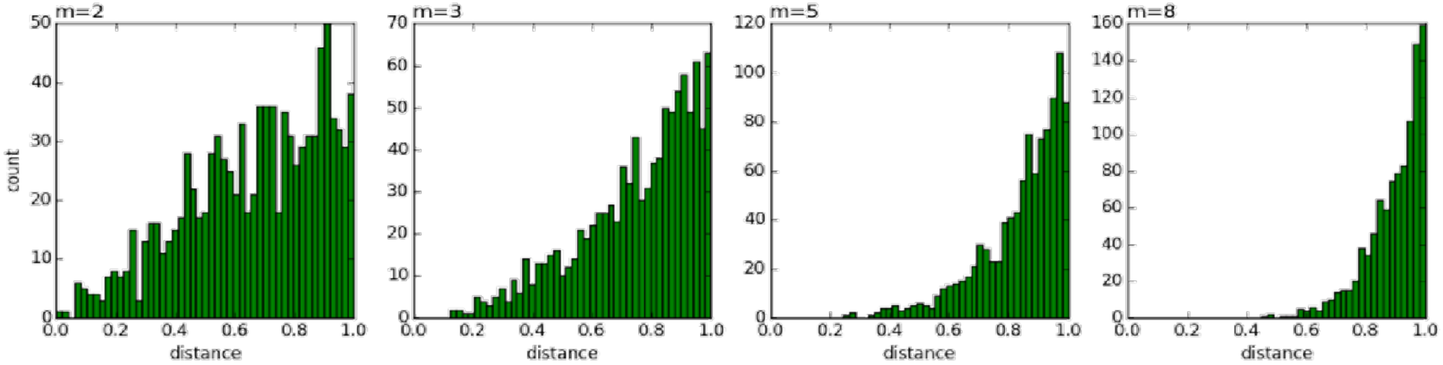
，Pj|i 越大

條件機率非對稱性：

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

Pj|i != Pi|j 且 Qi|j != Qj|i 且高維空間中兩點相距較遠時，
若映射到低維空間距離較近，反而得到很低的懲罰，有問題！

問題2



Crowding Problem(低維空間問題)：

想像一顆以點Xi為中心，半徑r的m維球，其他點與Xi隨維度增大距離越不均衡，若壓縮到低維，便會出現擠壓問題

問題怎麼解決？

t-SNE

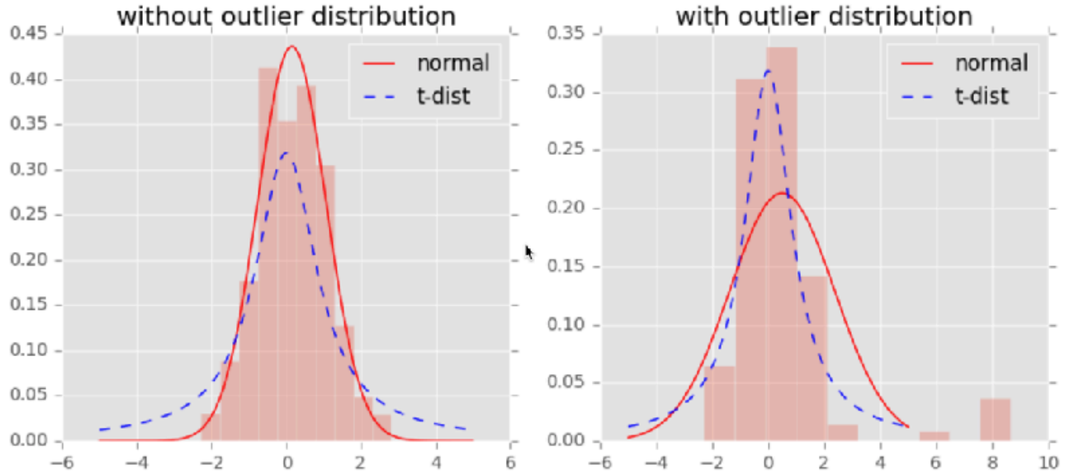
問題1　Pj|i != Pi|j 且 Qi|j != Qj|i

解法　以聯合機率分佈得到 Pij = Pji

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

問題2　Crowding Problem：壓縮到低維，會出現擠壓問題

解法　t就是T分佈，對異常點較不敏感，低維空間時取代高斯分佈，自由度越大，趨近於常態分佈




不如一槍打死我

t-SNE 將高維的數據用高斯分佈的機率密度函數近似，再將低維數據的部分使用t分佈的方式來近似，接著使用KL距離計算相似度，最後再以梯度下降求最佳解 。

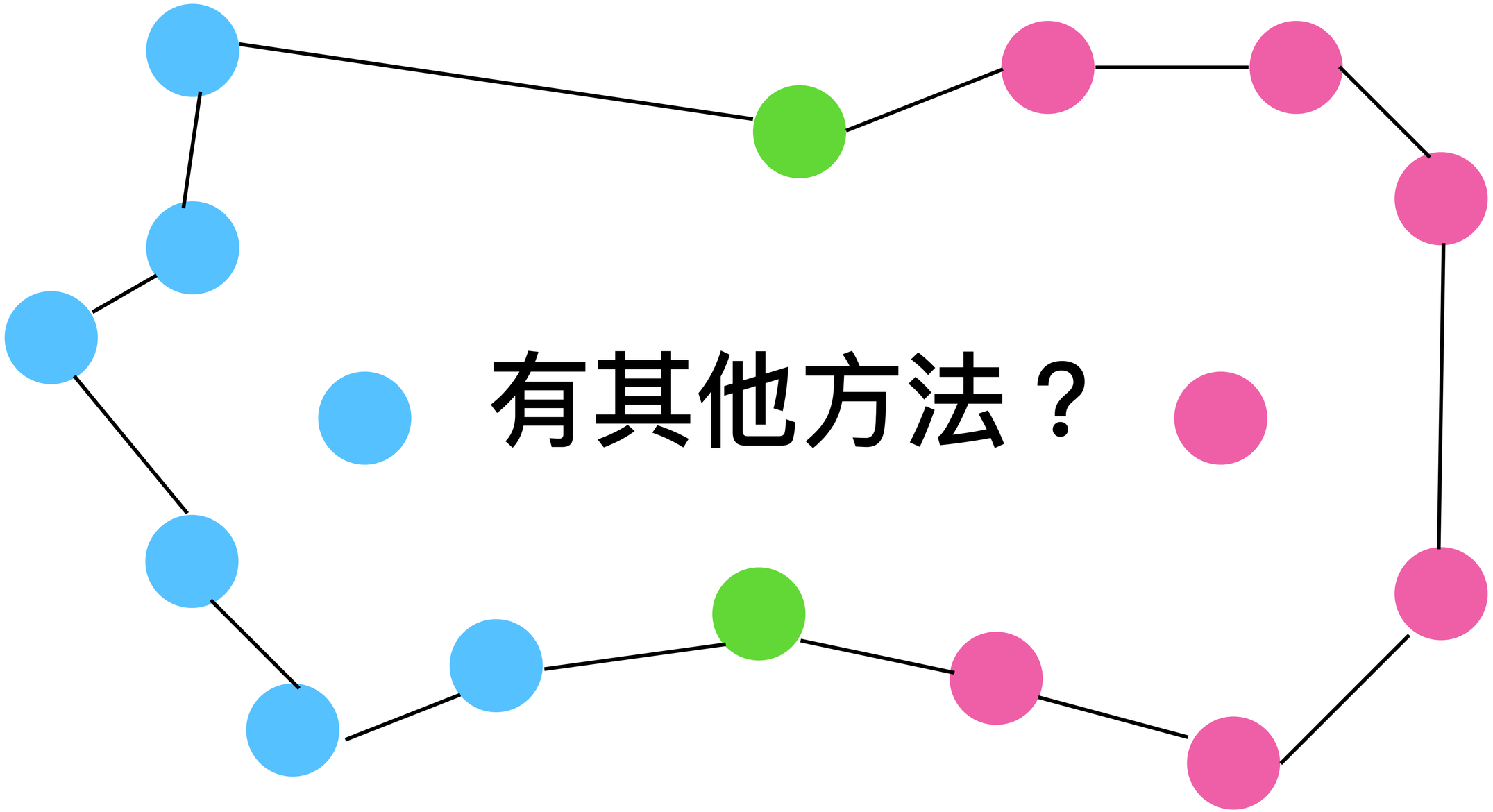
t-SNE的限制

降維是透過機率分佈的方式，無法應用

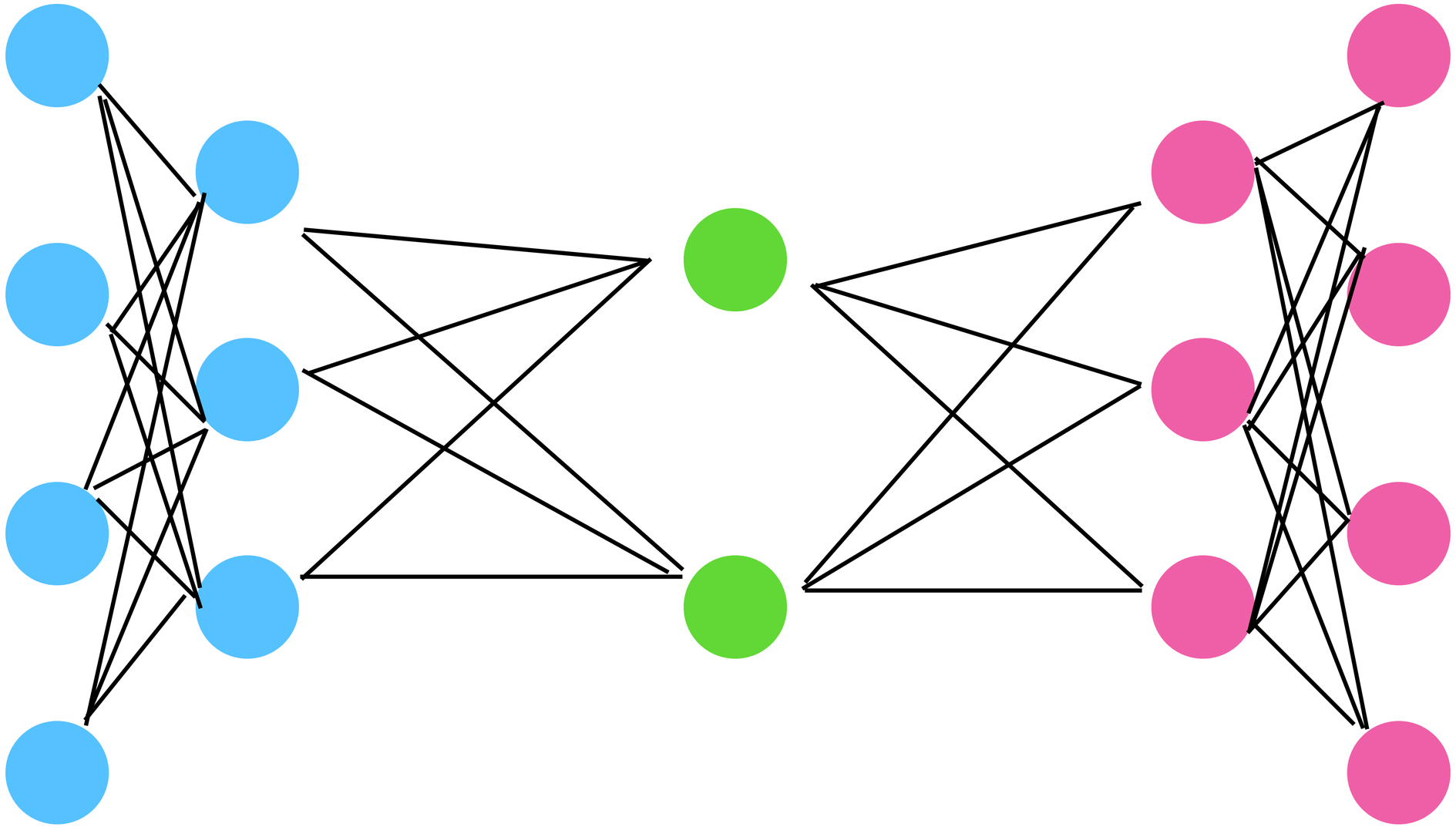
在全新資料集，通常僅拿來做成可視化




	PCA	t-SNE
Definitions	• Calculated through eigenvalues and eigenvectors. • Linear reduction.	• Calculated through probability distribution. • Also for non-linear reduction.
Stochasticity	• PCA is deterministic	• t-SNE is not.
Application to new Data	• Eigenvectors offer a new axes system what can be used to project new data.	• Learned by directly moving the data across the low dimensional space.
Time	• Fast.	• Usually need more time.


有其他方法？

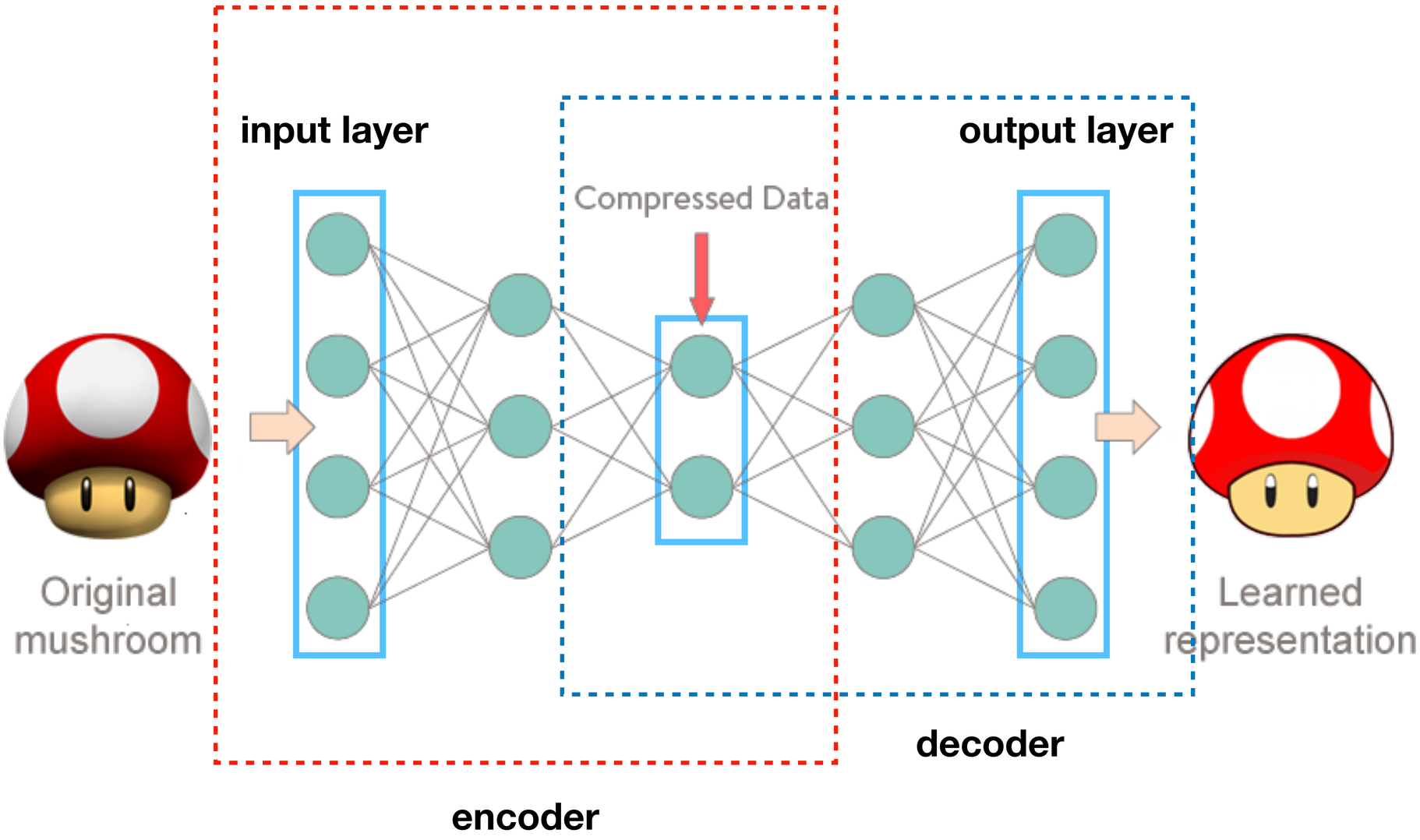
Autoencoder



什麼是 (AE)
Autoencoder ?

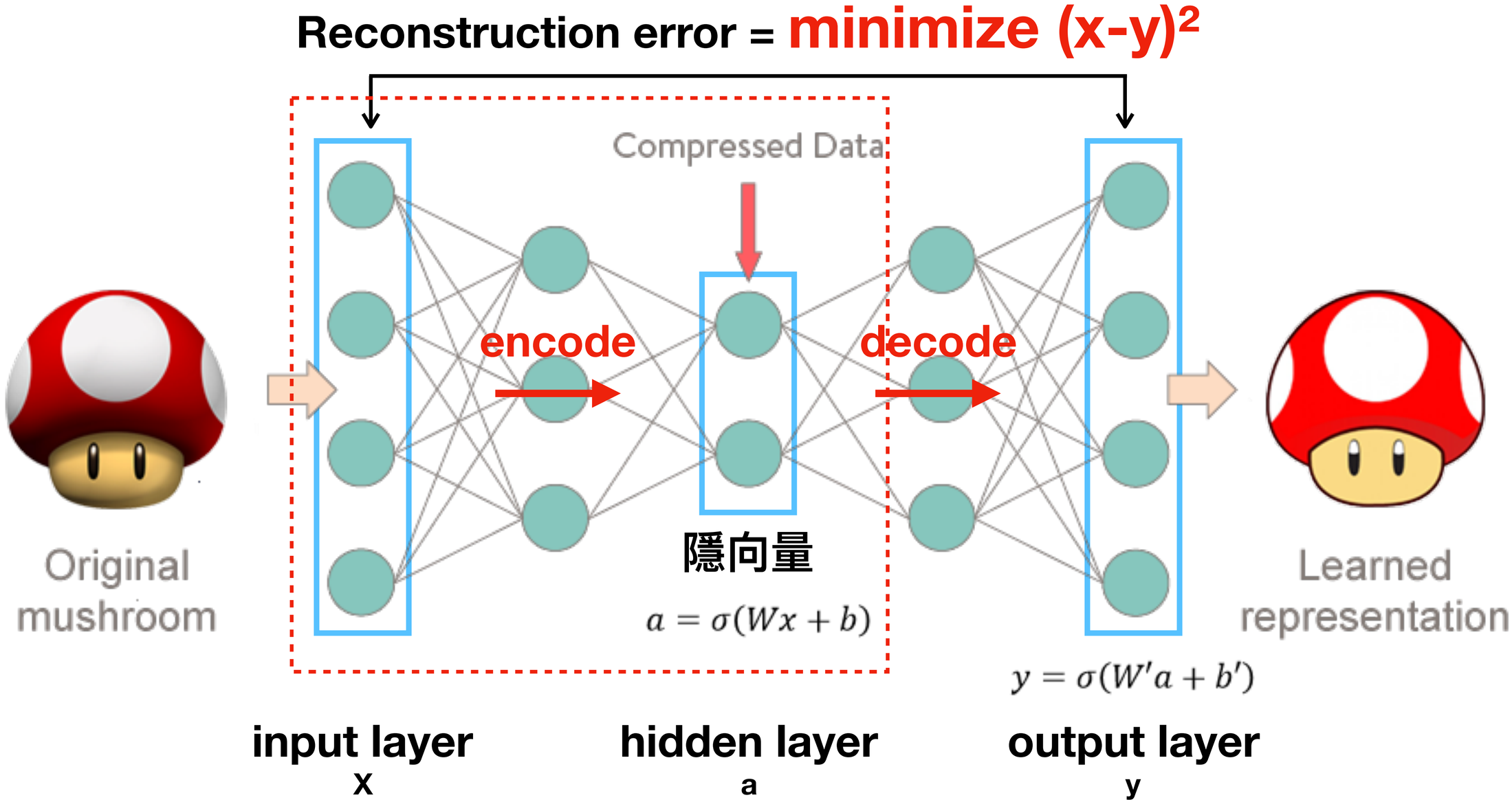
一種無監督式算法，利用反向傳播算法，使得目標值 = 輸入值

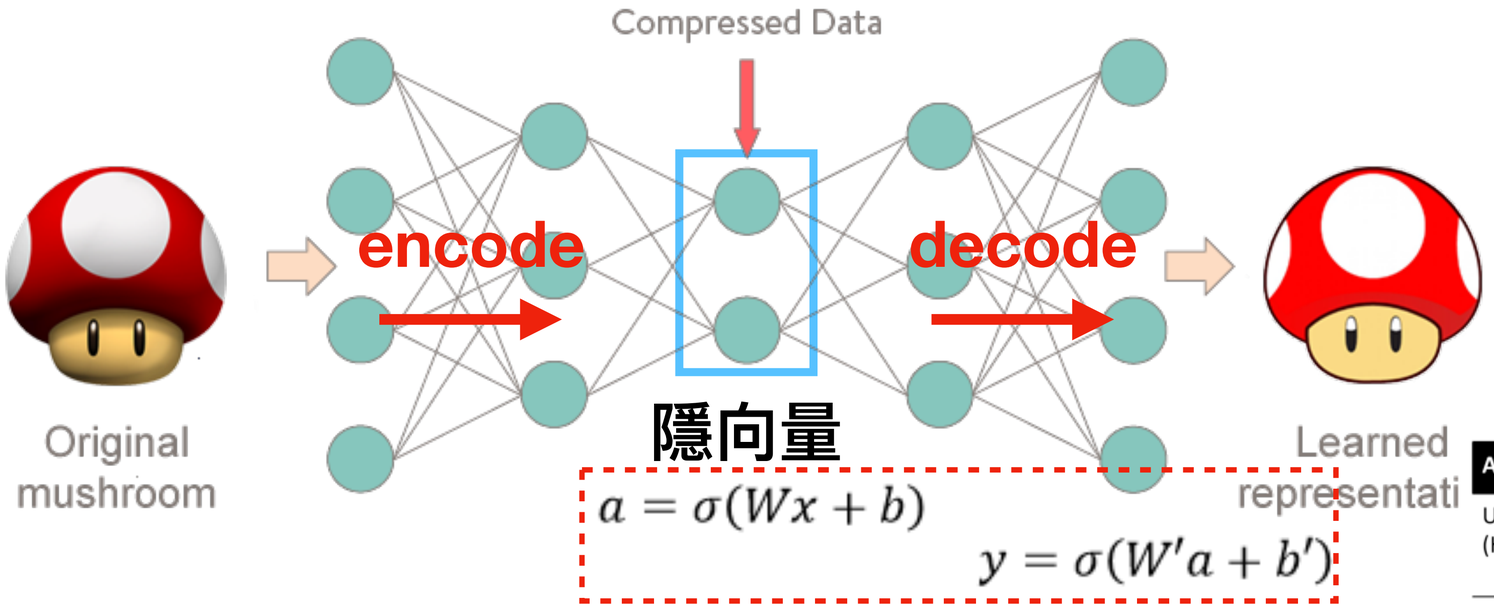
Autoencoder = encoder(降維) + decoder

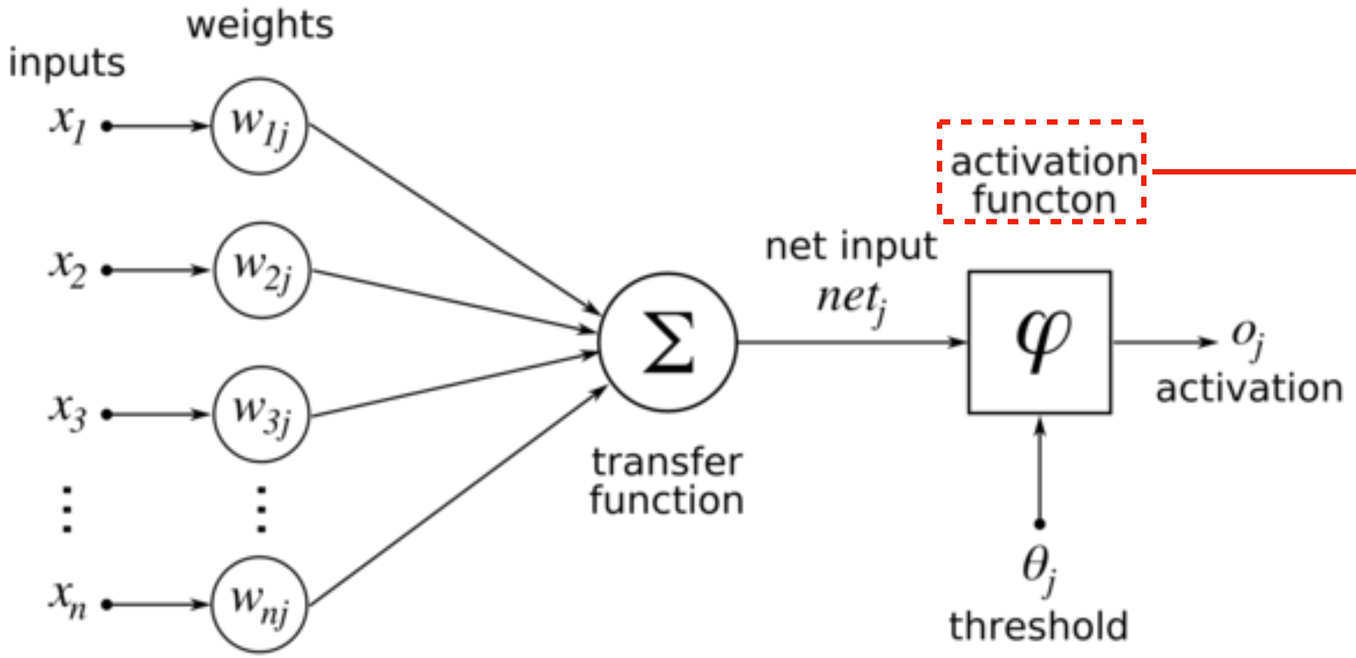


從有碼到無碼的還原過程

隱向量 = 精華層


Reconstruction error = minimize (x-y)²
Compressed Data
encode
decode
隱向量
$a = \sigma(Wx + b)$
$y = \sigma(W'a + b')$
Original mushroom
Learned representation
input layer
x
hidden layer
a
output layer
y





Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}. \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus	$\phi(z) = \ln(1 + e^z)$	Multi-layer Neural Networks	

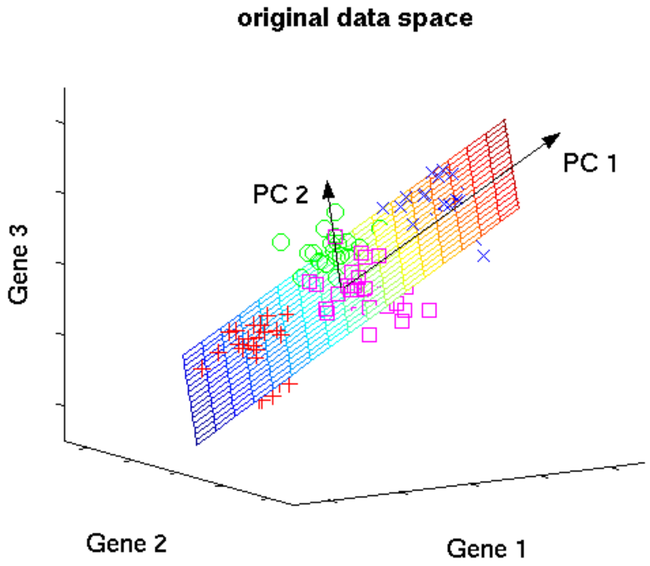
Copyright © Sebastian Raschka 2016 (http://sebastianraschka.com)

你發現什麼了嗎？

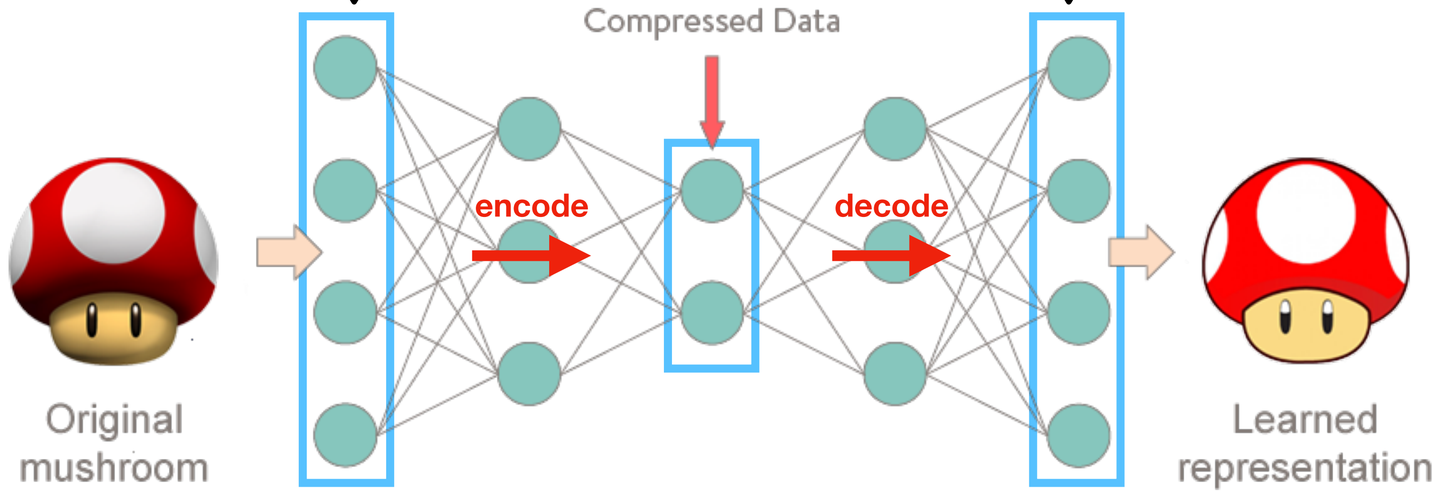
PCA: given M<d. Find $(u_1..u_M)$

that minimizes $E_M = \sum_{k=1}^{d} \left\| x_k - \hat{x}_k \right\|_2^2$

where $\hat{x}_k = \bar{x} + \sum_{i=1}^{M} z_i^k u_i$



Reconstruction error = minimize $(x-y)^2$



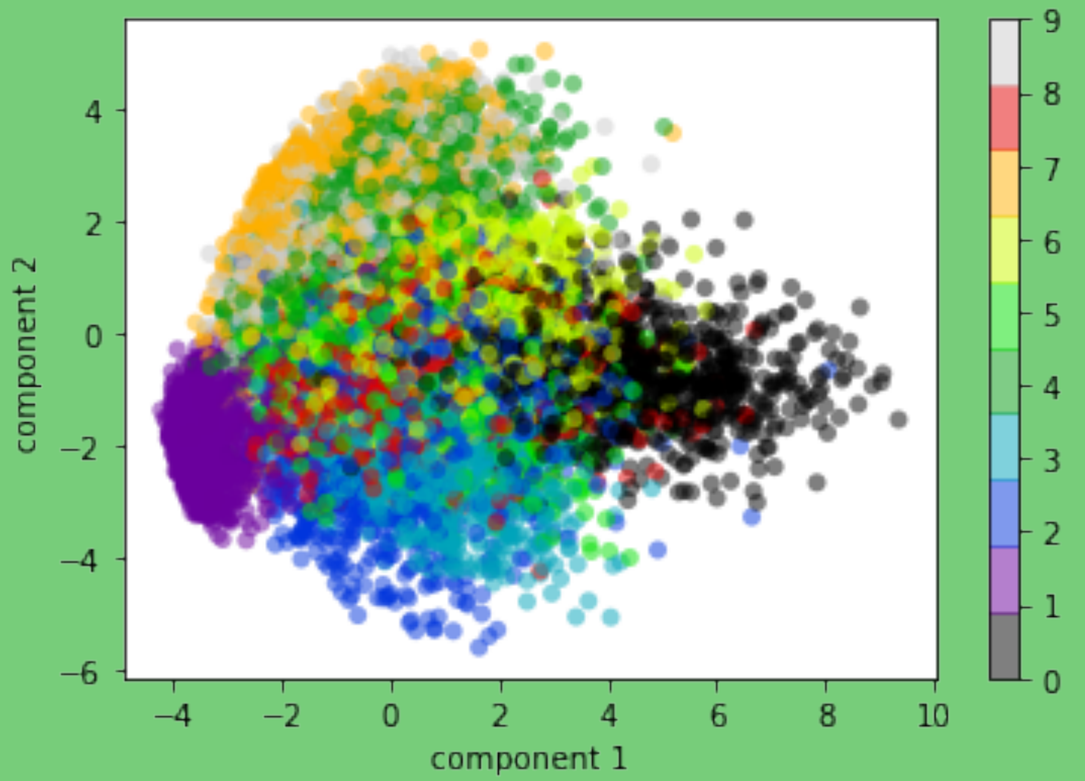
input layer x

hidden layer a

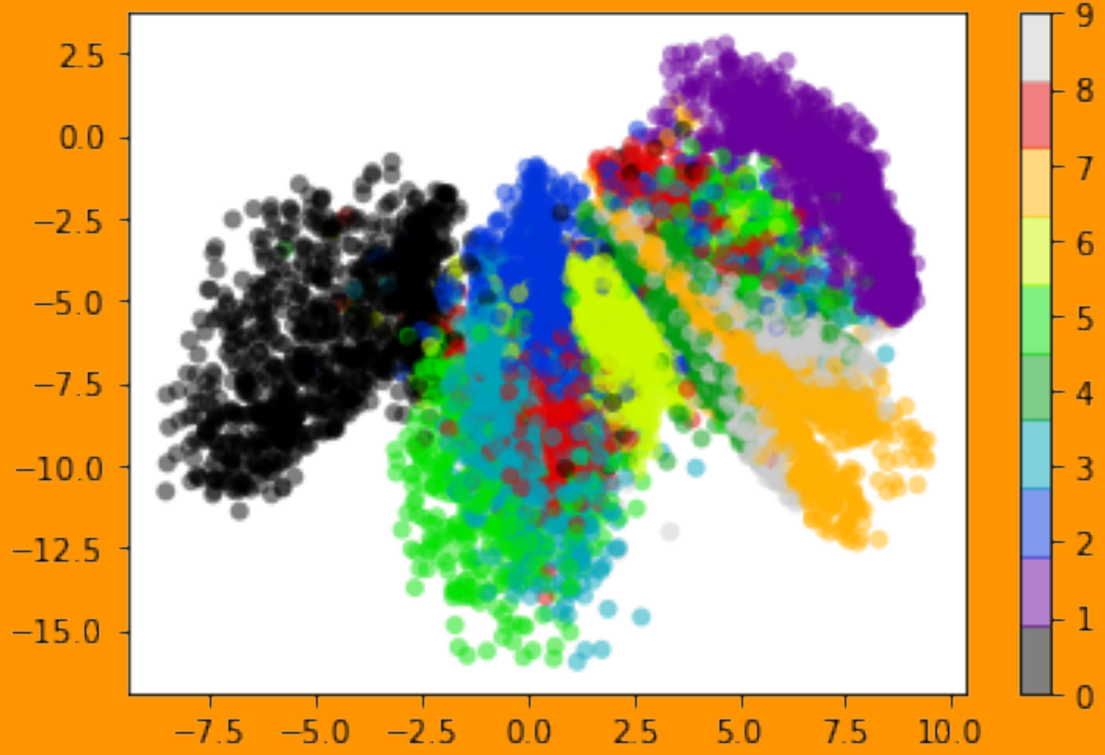
output layer y

當autoencoder只有一個隱含層且為線性的時候，其原理相當於PCA

PCA



Autoencoder

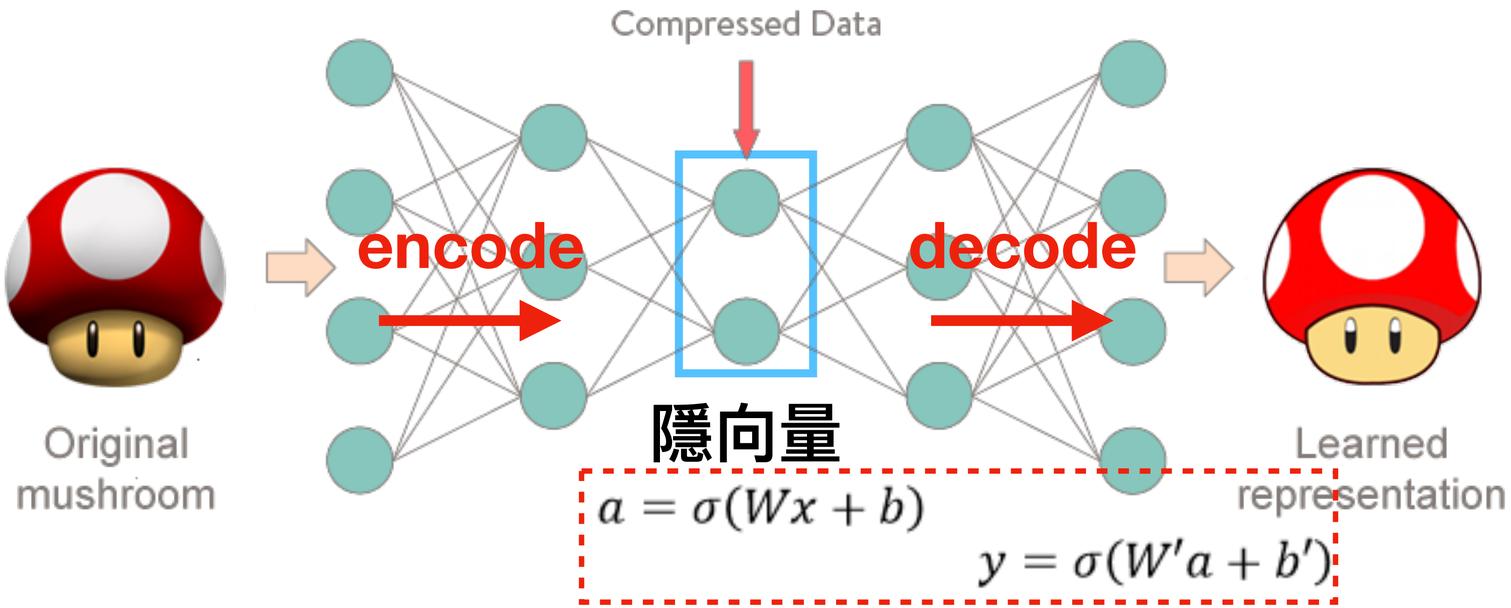


Autoencoder應用

Feauture extraction & reduction with AutoEncoder

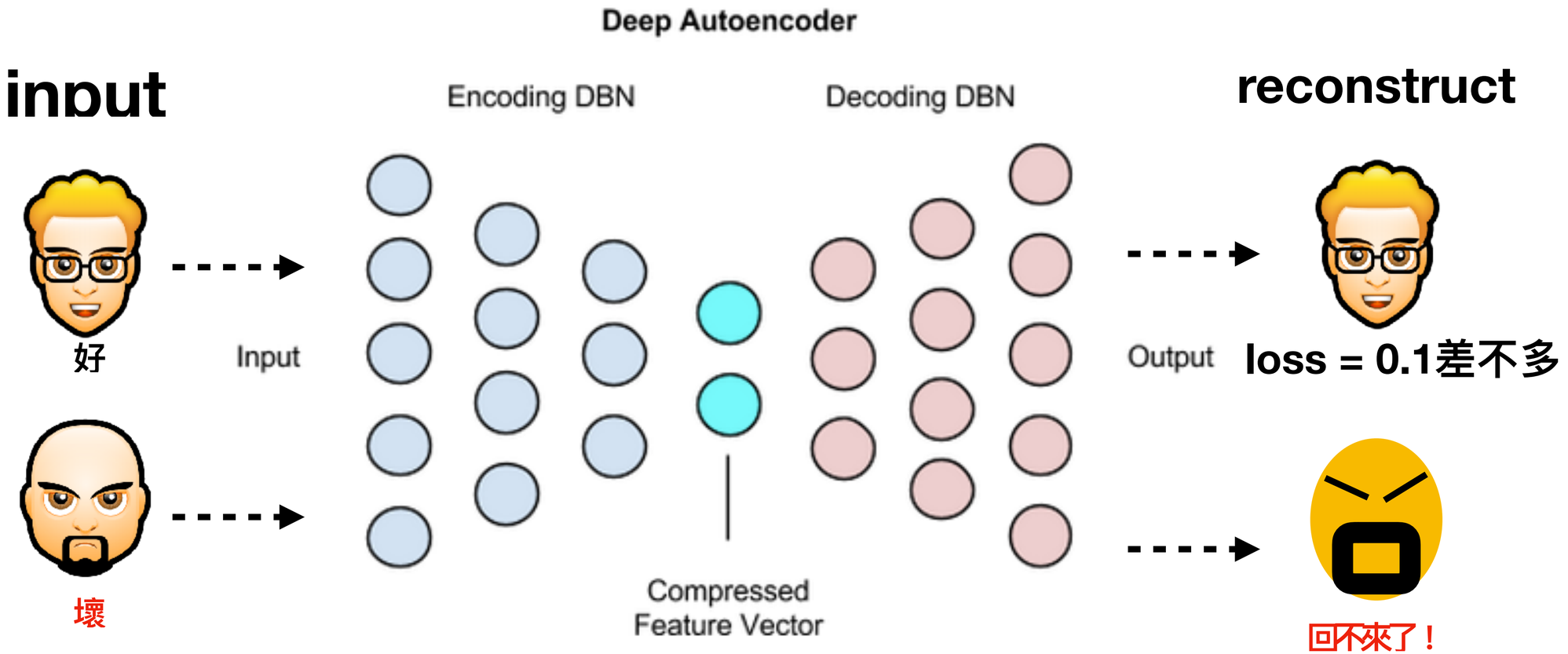
從高維度訊息量，提取最具信息量的關鍵特徵，達到降維的效果

Pre-Training with AutoEncoder



Pre-Training對初始化網絡訓練出更好的權重(W)，方便後續模型訓練

Anomaly Detection using AutoEncoder

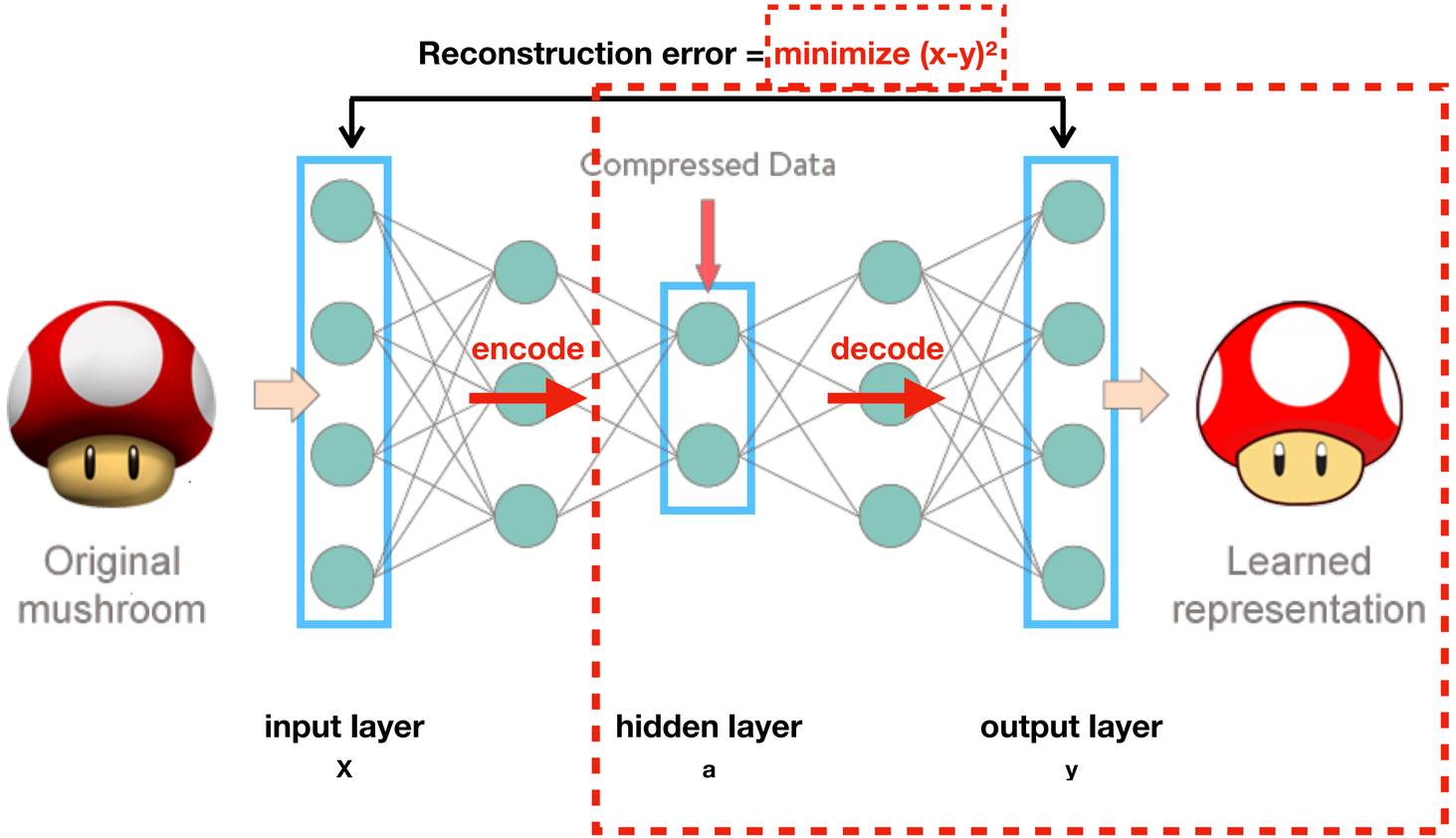


用好人訓練模型，

loss太大代表壞人

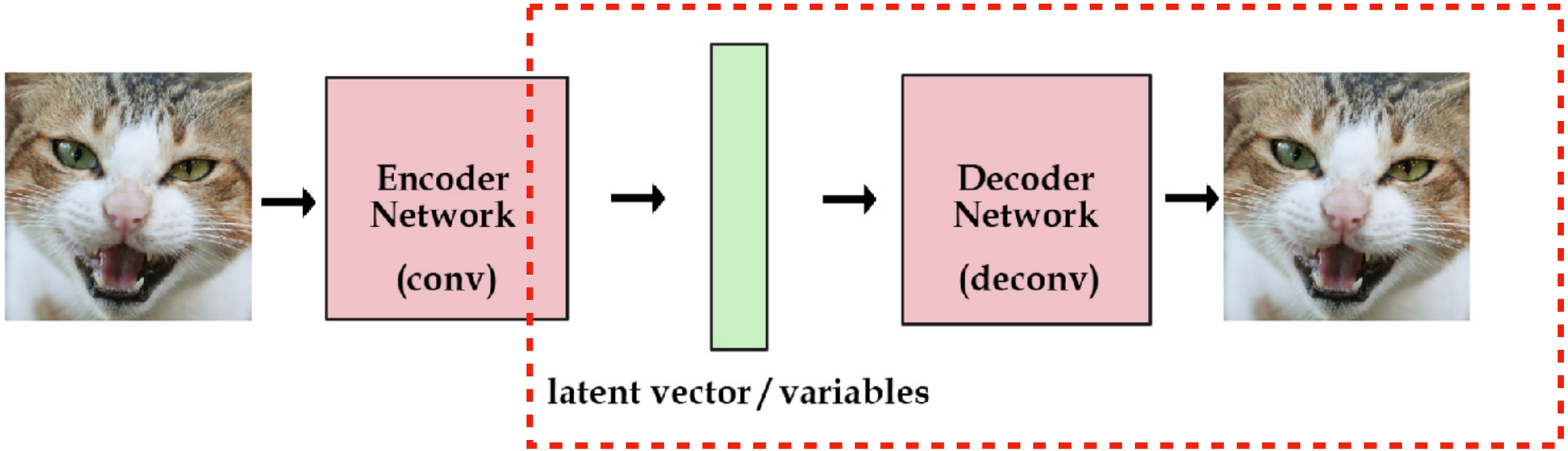

你真的以為我那麼好騙嗎

回憶



Decoder是一種還原過程

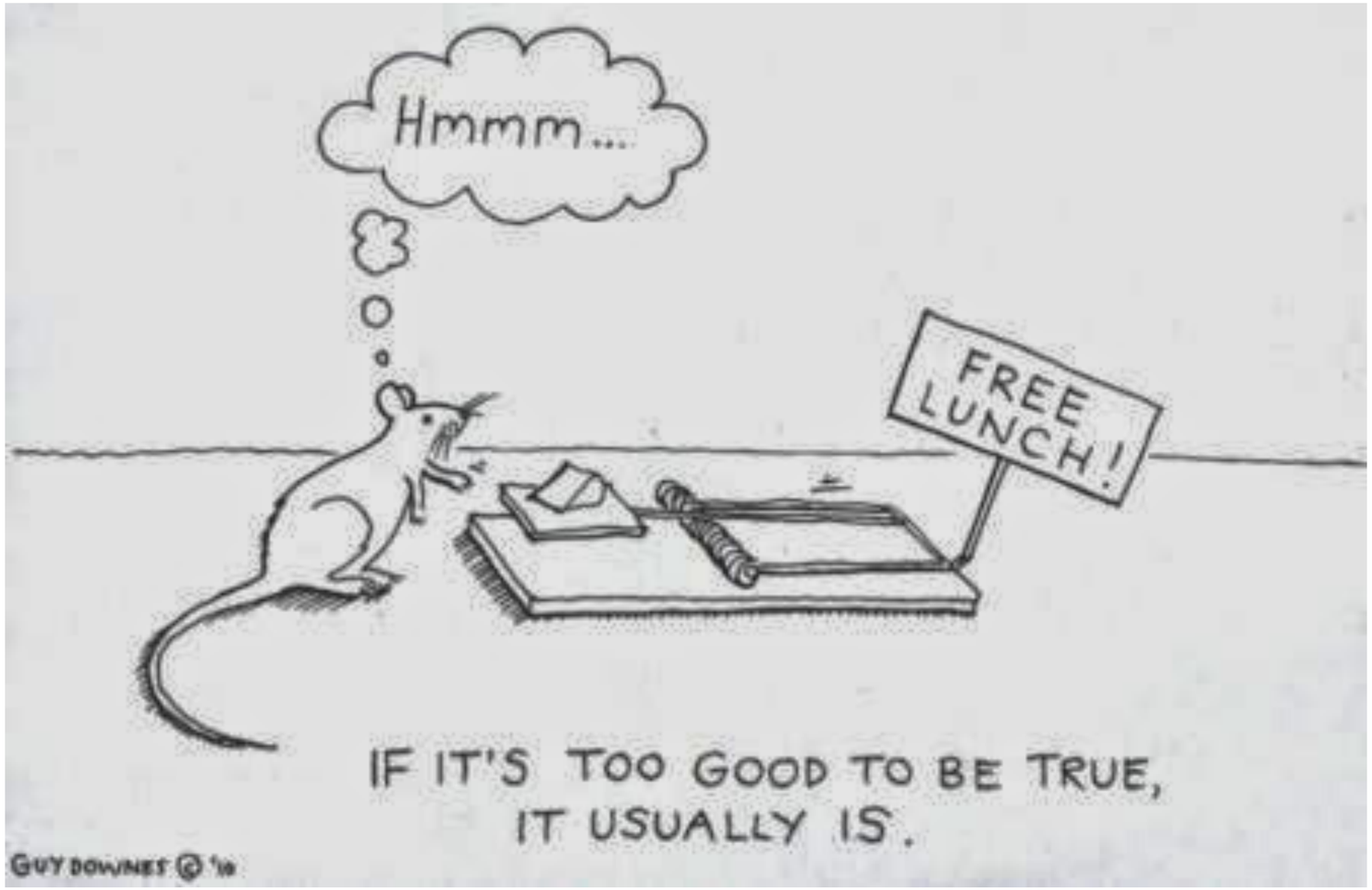
? Decode新圖片很糟



Autoencoder訓練隱向量還原效果有時很差

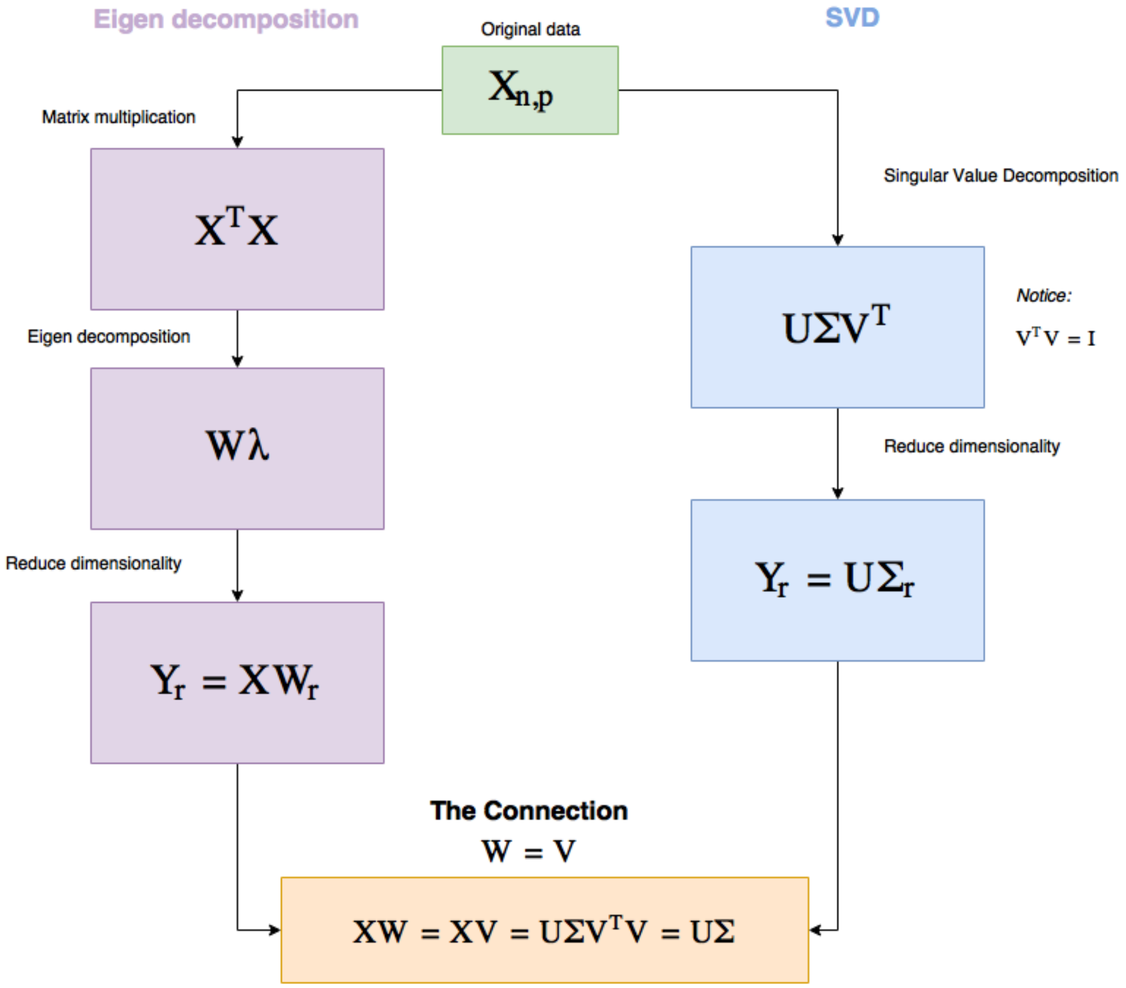
(因為學不到資料的機率分佈)

數據缺失怎麼辦啊

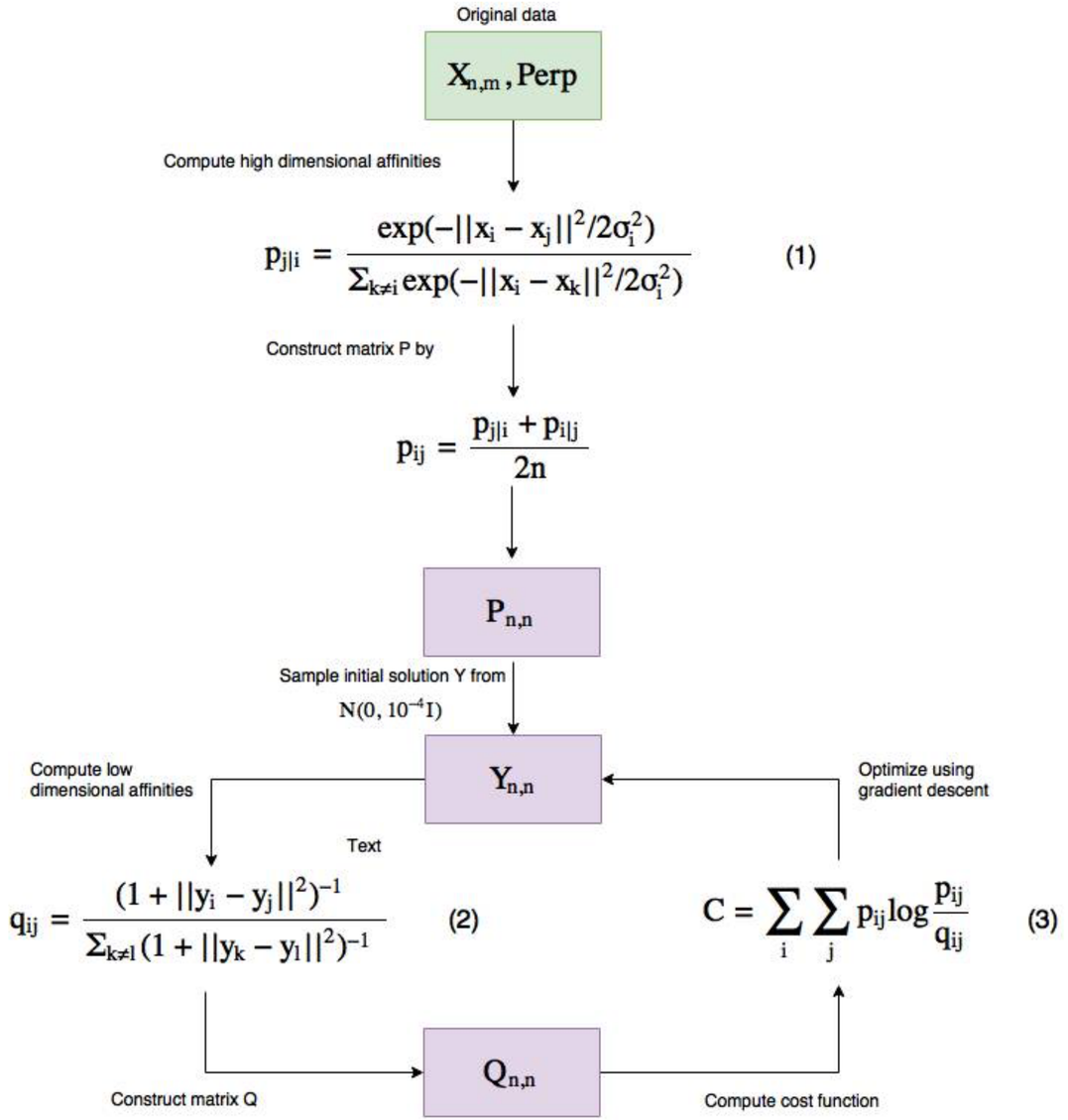

Hmmm...
FREE LUNCH!
IF IT'S TOO GOOD TO BE TRUE, IT USUALLY IS.
GUY DOWNES © '18

且待下回揭暁

PCA


Eigen decomposition
Original data
SVD
$X_{n,p}$
Matrix multiplication
Singular Value Decomposition
X^TX
$U\Sigma V^T$
Notice:
$V^TV = I$
Eigen decomposition
$W\lambda$
Reduce dimensionality
Reduce dimensionality
$Y_r = XW_r$
$Y_r = U\Sigma_r$
The Connection
$W = V$
$XW = XV = U\Sigma V^TV = U\Sigma$

t-SNE


Original data
$X_{n,m}$, Perp
Compute high dimensional affinities
$p_{j|i} = \frac{\exp(-||x_i - x_j||^2/2\sigma_i^2)}{\Sigma_{k\neq i}\exp(-||x_i - x_k||^2/2\sigma_i^2)}$ (1)
Construct matrix P by
$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
$P_{n,n}$
Sample initial solution Y from $N(0, 10^{-4}I)$
$Y_{n,n}$
Compute low dimensional affinities
Optimize using gradient descent
Text
$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\Sigma_{k\neq l}(1 + ||y_k - y_l||^2)^{-1}}$ (2)
$C = \sum_i \sum_j p_{ij}\log\frac{p_{ij}}{q_{ij}}$ (3)
$Q_{n,n}$
Construct matrix Q
Compute cost function