南京大学
NANJING UNIVERSITY

人工智能导论

无监督学习
(unsupervised learning)

郭兰哲

南京大学 智能科学与技术学院

Homepage: www.lamda.nju.edu.cn/guolz

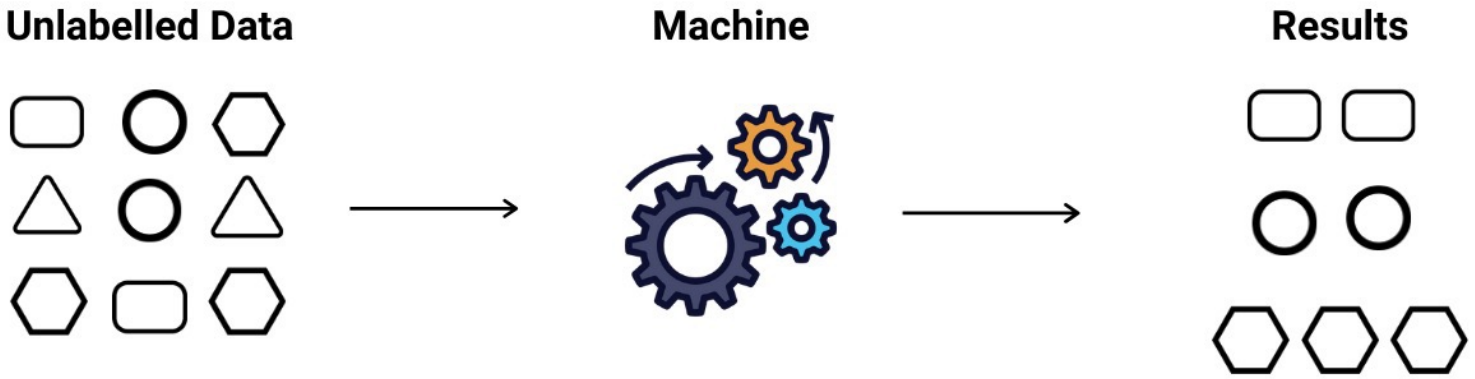
Email: guolz@nju.edu.cn

回顾：有监督学习(supervised learning)

- 常用评价指标：分类正确率，回归均方误差
- 近邻学习器：懒惰学习的代表，无需训练
- 决策树：不断选择属性划分节点建树
- 贝叶斯分类器：生成式方法的代表，估计联合分布再计算后验分布
- 线性回归：最小二乘法的闭式解
- 对数几率回归：引入sigmoid函数解决分类问题
- 支持向量机：大间隔、核函数

无监督学习

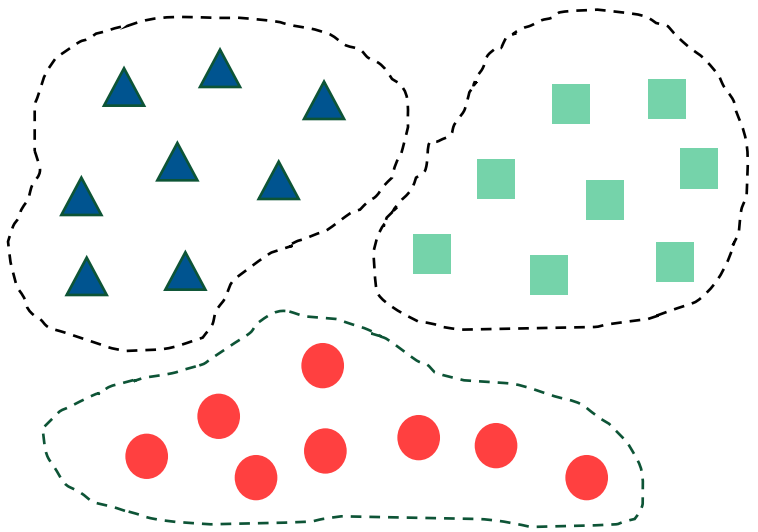
无监督学习：所有训练样本均没有标注



聚类(clustering)

在“无监督学习”任务中研究最多、应用最广

目标：将数据样本划分为若干个通常不相交的“簇”(cluster)



既可以作为一个单独过程用于找寻数据内在的分布结构，也可作为分类等其他学习任务的前驱过程

聚类的评价指标

聚类的“好坏”不存在绝对标准

the goodness of clustering depends on the opinion of the user

基本原则：

- “簇内相似度”（intra-cluster similarity）高，且
- “簇间相似度”（inter-cluster similarity）低

K均值聚类（K-Means）

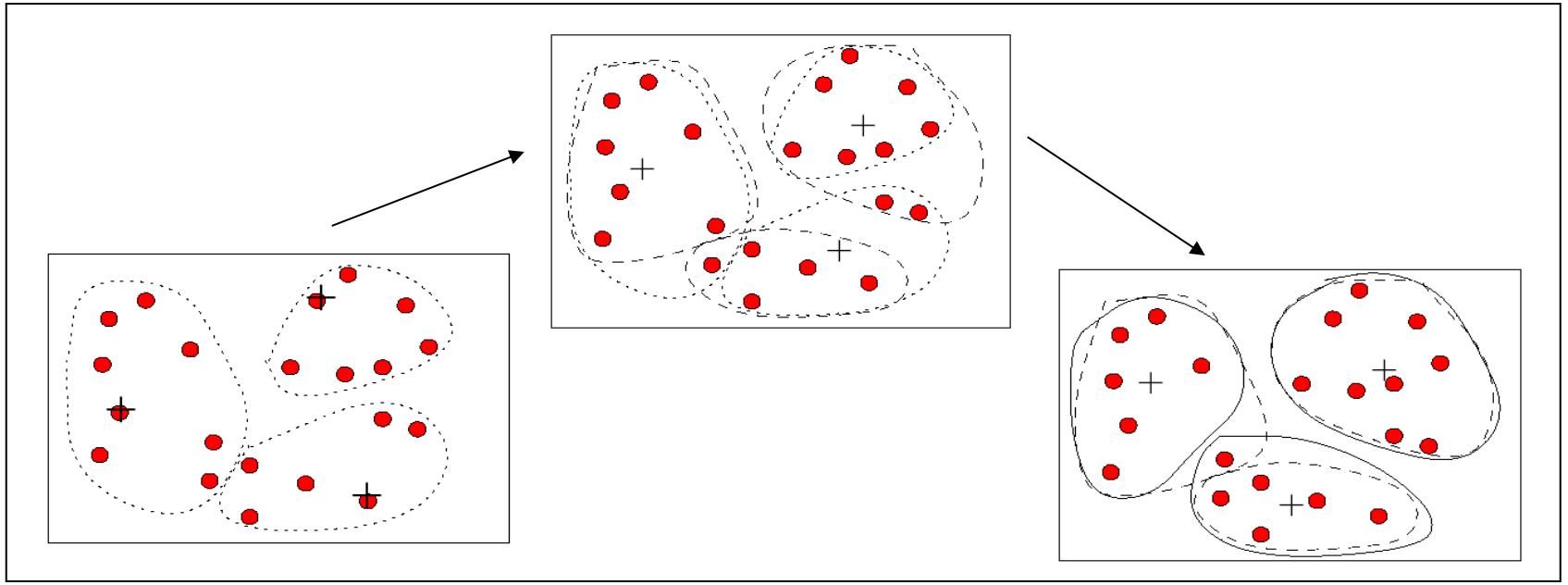
每个簇以该簇中所有样本点的“均值”表示

Step1：随机选取k个样本点作为簇中心

Step2：将其他样本点根据其与簇中心的距离，划分给最近的簇

Step3：更新各簇的均值向量，将其作为新的簇中心

Step4：若所有簇中心未发生改变，则停止；否则执行 Step 2



K均值聚类示例

例子：给定含有5个样本的集合，用K均值聚类将其聚成2类

$$X = \begin{bmatrix} 0 & 0 & 1 & 5 & 5 \\ 2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

（1）选择两个样本点作为类中心，(0,2), (0, 0)

（2）通过计算距离，第一个簇为{1,5}、第二个簇为{2,3,4}

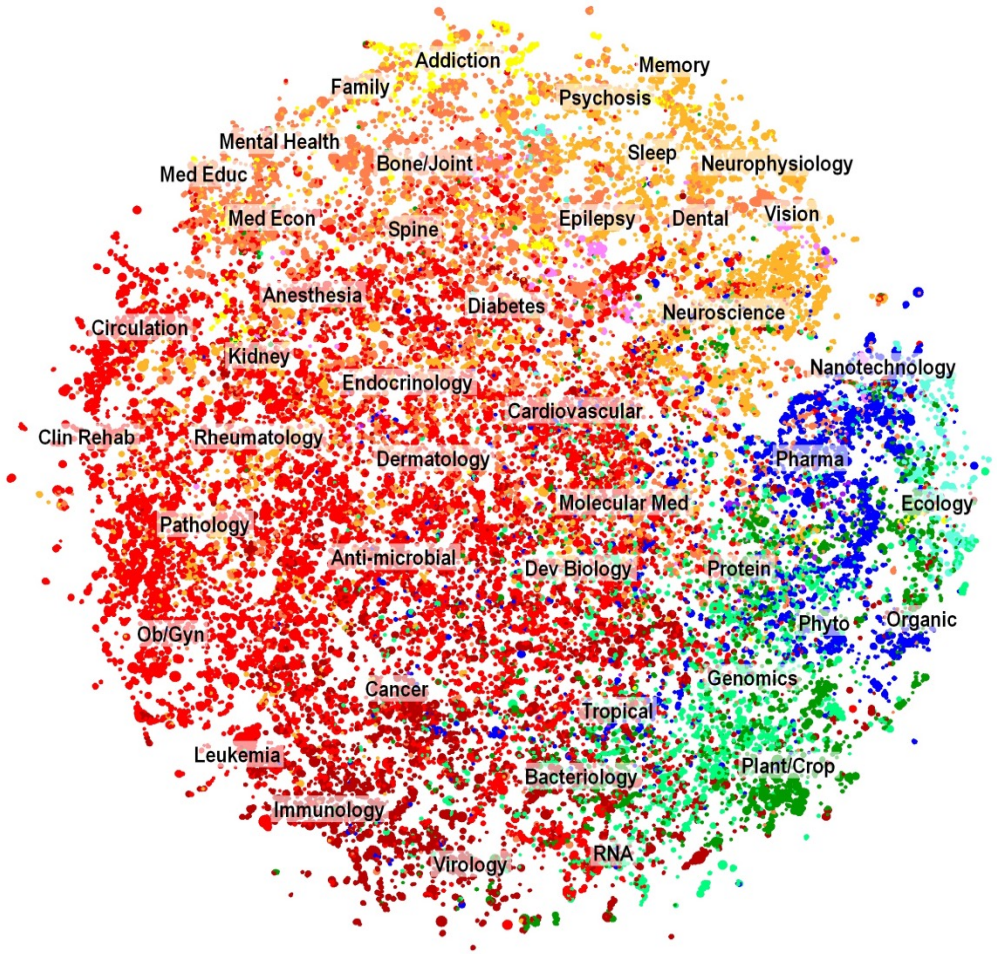
（3）计算新的类中心：(2.5,2), (2, 0)

（4）新的簇：第一个簇为{1,5}、第二个簇为{2,3,4}

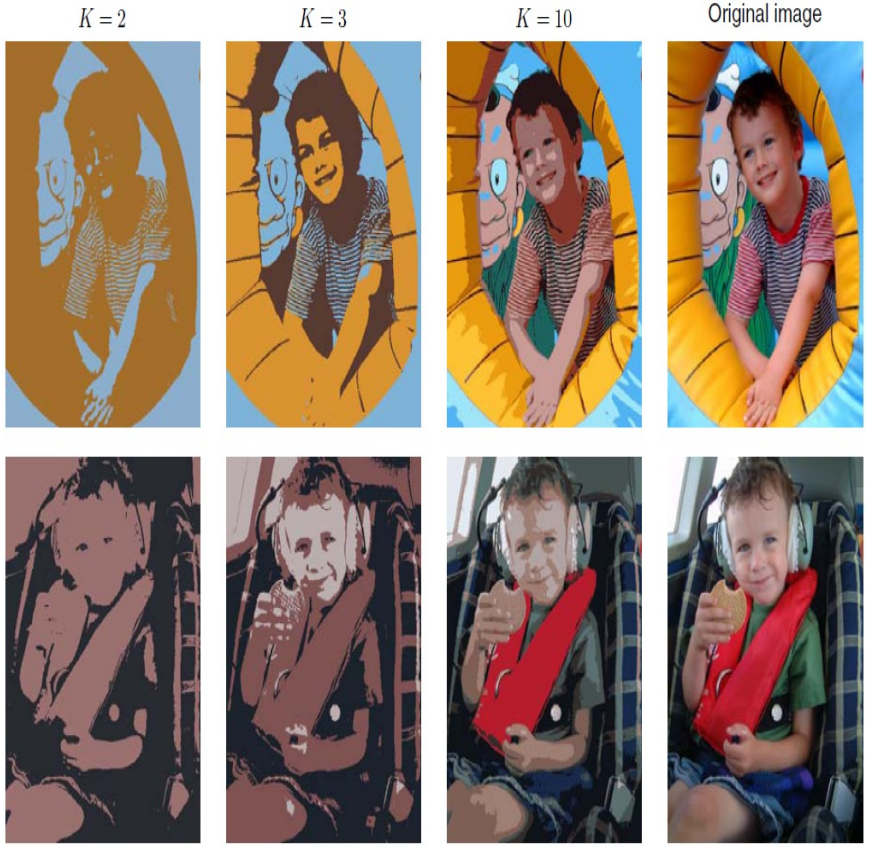
K均值聚类的不足

- 需要事先确定聚类数目，很多时候我们并不知道数据应被聚类的数目
- 需要初始化聚类质心，初始化聚类中心对聚类结果有较大的影响
- 算法是迭代执行，时间开销非常大
- 距离计算往往假设数据每个维度之间的重要性是一样的

K-Means的应用



文本分类：将200多万篇论文聚类到29,000个类别，包括化学、工程、生物、传染疾病、生物信息、脑科学、社会科学、计算机科学等及给出了每个类别中的代表单词



色彩压缩：每个簇的颜色变为同一种

使用sklearn实现

```
class sklearn.cluster.KMeans(n_clusters=8, *, init='k-means++', n_init='warn', max_iter=300, tol=0.0001, verbose=0, random_state=None, copy_x=True, algorithm='lloyd')
```

```
>>> from sklearn.cluster import KMeans
>>> import numpy as np
>>> X = np.array([[1, 2], [1, 4], [1, 0],
...               [10, 2], [10, 4], [10, 0]])
>>> kmeans = KMeans(n_clusters=2, random_state=0, n_init="auto").fit(X)
>>> kmeans.labels_
array([1, 1, 1, 0, 0, 0], dtype=int32)
>>> kmeans.predict([[0, 0], [12, 3]])
array([1, 0], dtype=int32)
>>> kmeans.cluster_centers_
array([[10.,  2.],
       [ 1.,  2.]])
```

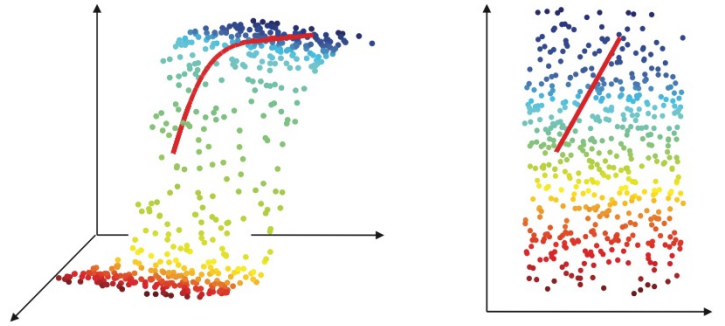
降维（dimension reduction）

- 机器学习方法在高维数据下面临数据样本稀疏、距离计算困难等问题，被称为维数灾难(curse of dimensionality)

⇨ 降维（将高维样本降到低维）

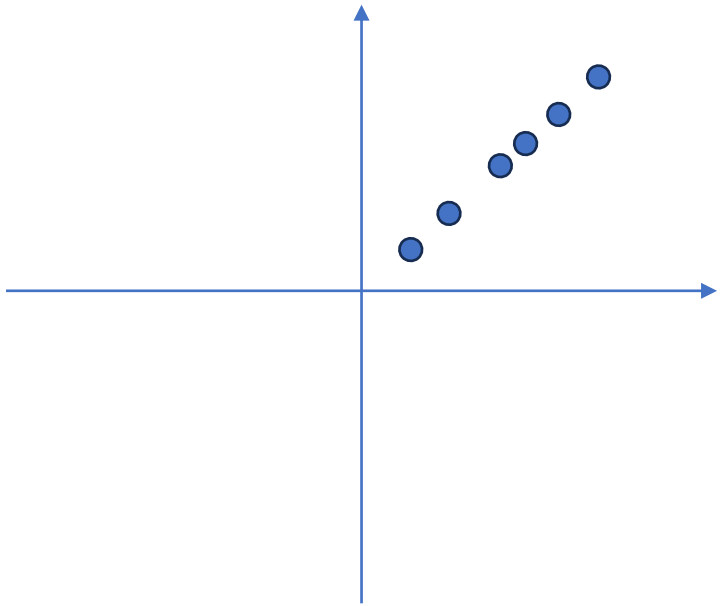
为什么能进行降维？

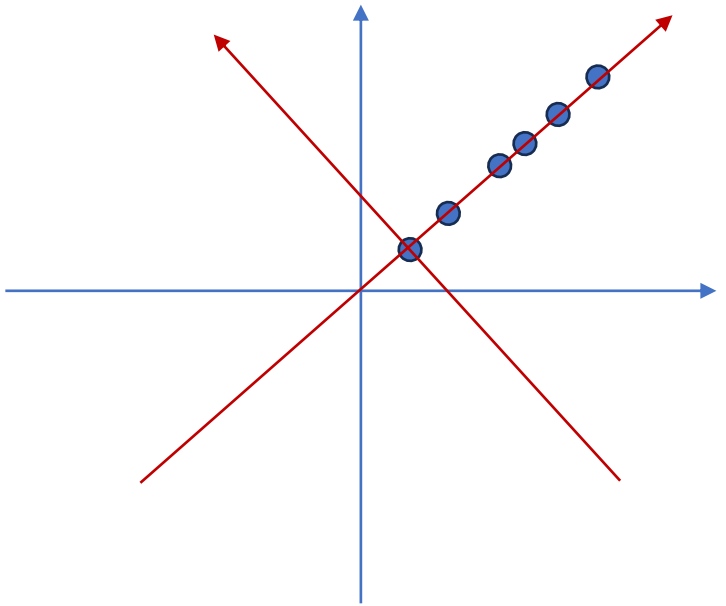
数据样本虽是高维的，但与学习任务密切相关的也许仅是某个低维空间，即高维空间中的一个低维"嵌入"(embedding)



(a) 三维空间中观察到的样本点　　(b) 二维空间中的曲面

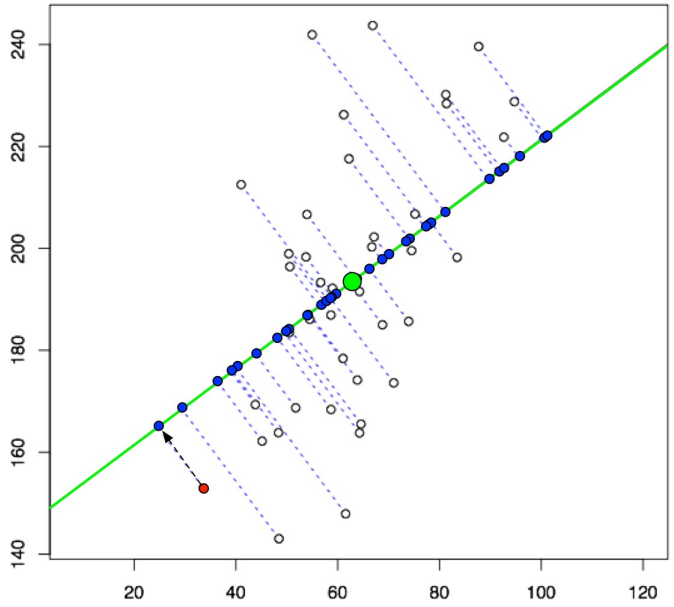
主成分分析（PCA, Principal Component Analysis）





PCA：找坐标系

主成分分析（PCA, Principal Component Analysis）



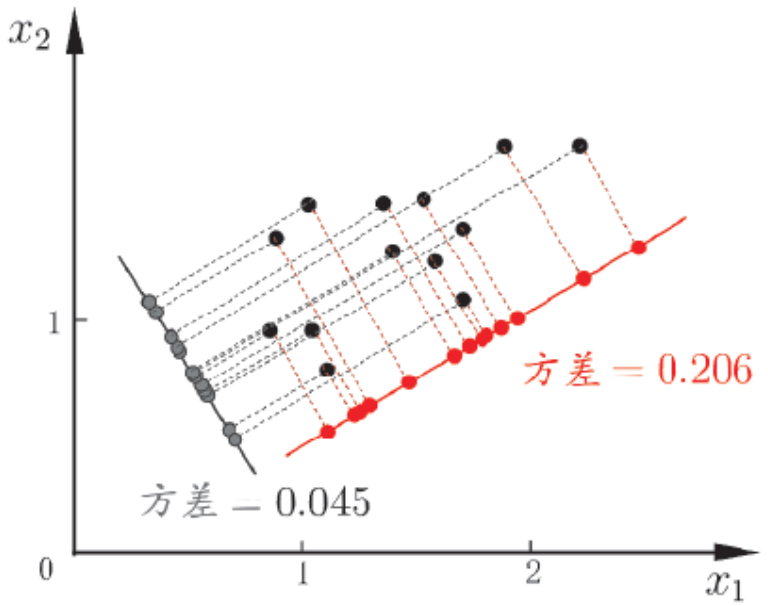
正交属性空间中的样本点，如何使用一个超平面对所有样本进行恰当的表达？

若存在这样的超平面，那么它大概应具有这样的性质：

- 最大可分性：
  - 样本点在这个超平面上的投影能尽可能分开

主成分分析（PCA, Principal Component Analysis）

样本点 $\boldsymbol{x}_i$ 在新空间中超平面上的投影是 $\mathbf{W}^{\mathrm{T}}\boldsymbol{x}_i$，若所有样本点的投影能尽可能分开，则应该使得投影后样本点的方差最大化



投影后样本点的方差是 $\sum_i \mathbf{W}^{\mathrm{T}}\boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}\mathbf{W}$

于是：

$$\begin{aligned} \max_{\mathbf{W}} \quad & \mathrm{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I}. \end{aligned}$$

等价于：

$$\begin{aligned} \min_{\mathbf{W}} \quad & -\mathrm{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I}. \end{aligned}$$

主成分分析（PCA, Principal Component Analysis）

$$\begin{aligned} \max_{\mathbf{W}} \quad & \mathrm{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I}. \end{aligned}$$

使用拉格朗日乘子法可得

$$\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W} = \lambda\mathbf{W}.$$

只需对矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 进行特征值分解，并将求得的特征值排序：$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$，再取前 d' 个特征值对应的特征向量构成 $\mathbf{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'})$，这就是主成分分析的解

主成分分析（PCA, Principal Component Analysis）

d' 的设置：

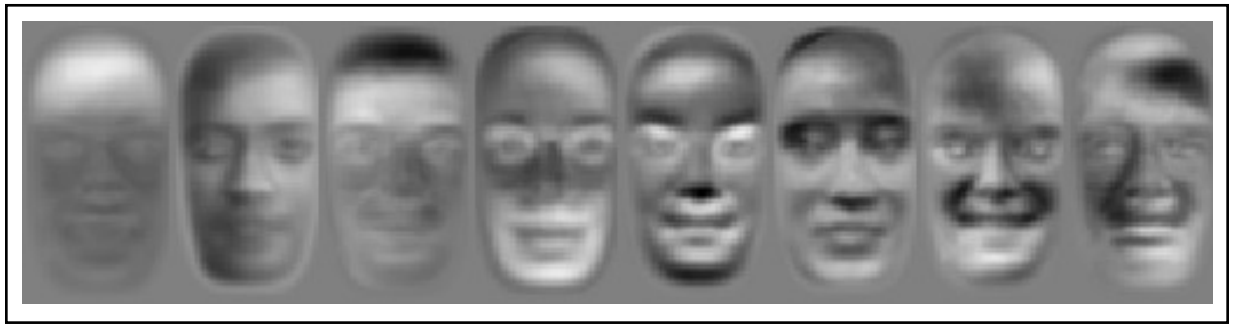
- 用户指定
- 在低维空间中对k近邻或其他分类器进行交叉验证
- 设置重构阈值，例如 t =95%，然后选取最小的 d' 使得 $\frac{\sum_{i=1}^{d'} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \geq t.$

Eignface

PCA 是最常用的降维方法，在不同领域有不同的称谓

例如在人脸识别中该技术被称为“特征脸”(eigenface)

因为若将前 d' 个特征值对应的特征向量还原为图像，则得到



使用sklearn实现

```
class sklearn.decomposition.PCA(n_components=None, *, copy=True, whiten
=False, svd_solver='auto', tol=0.0, iterated_power='auto', n_oversample
s=10, power_iteration_normalizer='auto', random_state=None)
```

In [1]:
```
import numpy as np
from sklearn.decomposition import PCA
X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
pca = PCA(n_components=2)
pca.fit_transform(X)
```

Out[1]:
```
array([[ 1.38340578,  0.2935787 ],
       [ 2.22189802, -0.25133484],
       [ 3.6053038 ,  0.04224385],
       [-1.38340578, -0.2935787 ],
       [-2.22189802,  0.25133484],
       [-3.6053038 , -0.04224385]])
```

In [2]:
```
pca = PCA(n_components=1)
pca.fit_transform(X)
```

Out[2]:
```
array([[ 1.38340578],
       [ 2.22189802],
       [ 3.6053038 ],
       [-1.38340578],
       [-2.22189802],
       [-3.6053038 ]])
```

小结

- 聚类：针对给定的样本，依据它们属性的相似度或距离，将其划分到若干个“簇”的数据分析问题
- 距离或相似度度量在聚类中起着重要作用
- K均值聚类的算法流程
- PCA的推导与几何意义