1. **聚类**

**1.1 K-Means：**

最普遍最简单的聚类算法，K-Means，其中的K是初始设定的分类的数量。

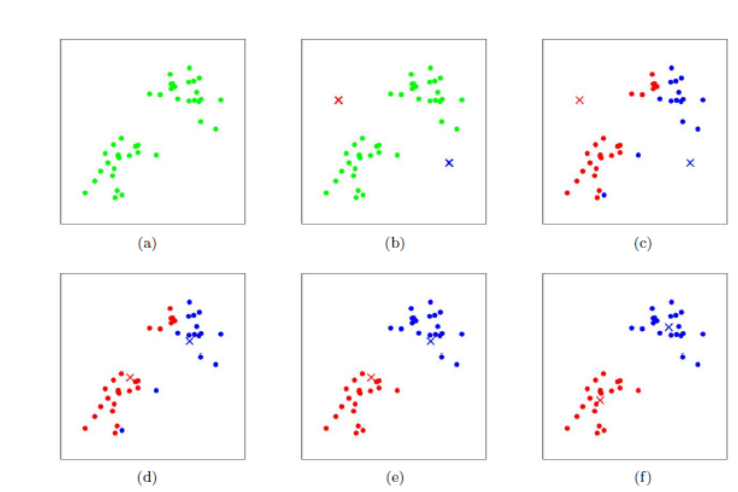
计算步骤：假设样本为，其中。

1. 随机选取K个聚类质心点。
2. 对每一个样本，计算其应该属于的类。

计算L2范数，

其中为类别。重新计算每一个类的聚类质心，求该类中所有样本的平均值，

1. 重复步骤2)，直到聚类质心不发生变化或变化很小



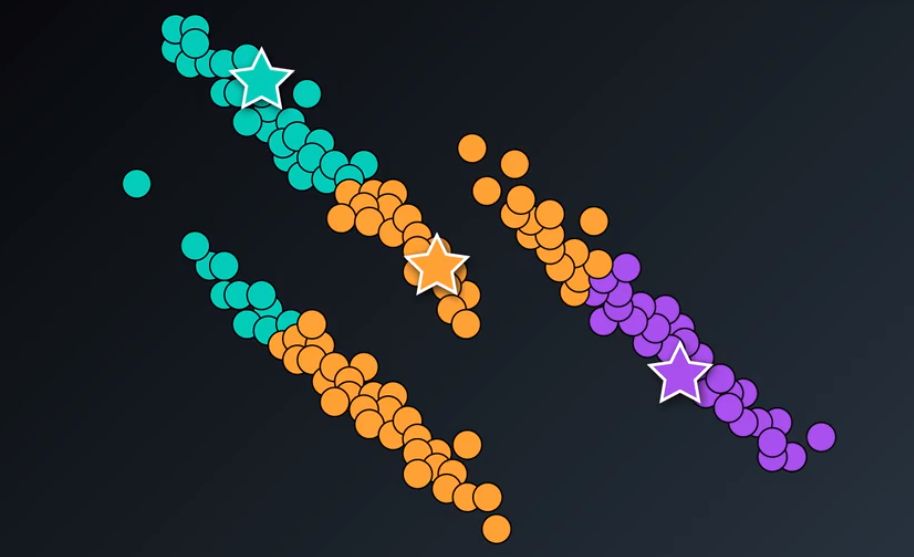
K值选取：肘部方法

<https://blog.csdn.net/qq_15738501/article/details/79036255>

K-Means的优点：算法简单，速度快，可解释性强，处理大量数据高效。

K-Means的缺点：

1. 相同的数据集和相同的分类数量下，K-Means得到的结果有可能完全不同，它非常依赖于初始聚类质心所处的位置。
2. 对离群值敏感。
3. 无法处理非球形簇，例如：



**1.2 层次聚类**：分为凝聚聚类和分裂聚类（很少用）

凝聚聚类：初始时将每个样本都看成一个类，然后根据规则进行合并，直到类的数目满足设定的值。

sklearn中的凝聚聚类分为：这里的距离指的是欧氏距离（L2范数）

<https://wenku.baidu.com/view/21c57ac7a32d7375a51780d3.html>

凝聚聚类根据不同的判别方式，又分为以下几种：

1. 单连接聚类（最近邻聚类）

第1步将每个样本都看成一个类，将距离最近的两个样本合并为1类；

第2步计算各个类之间的距离。设表示类，表示样本i和样本j的距离，表示类和之间的距离：两个类样本的最小距离，即认为只要两个类中有样本近似，就认为两个类可以合并为一个类，可能会破坏类的紧凑性



将类距离最近的两类合并，以此类推，直到类的数目满足设定的值

1. 全连接聚类（最远邻聚类）

全连接的聚类过程与单连接完全一样，也是将两个距离最近的类合并，不同的是类的距离的计算与单连接不同：两个类样本的最大距离，即认为两个类中所有样本都近似时，两个类才能合并，



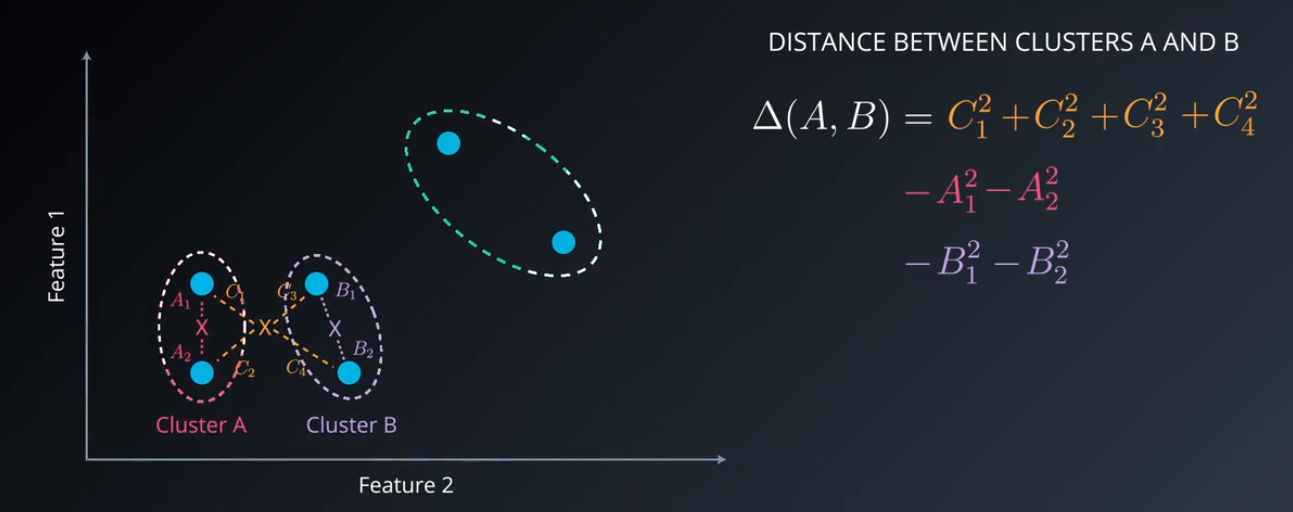
1. 组平均聚类

两个类的距离的计算：两个类的样本的平均平方距离



1. 离差平方和聚类

两个类的距离的计算：



其中为类A和类B中的每个样本到类A和类B的中心点的距离的平方，为类A中的每个样本到类A的中心点的距离的平方，为类B中的每个样本到类B的中心点的距离的平方。

层次聚类的优点：

1. 距离和规则的相似度容易定义，限制少
2. 不需要预先估算聚类数（sklearn的函数要指定聚类数，但层次聚类在原理上是将所有样本都聚成1类，所以不需要人为的判断数据集到底有几类）
3. 可以反映出类的层次关系

层次聚类的缺点：

1. 计算复杂度高
2. 对噪声和离群值很敏感

**DBSCAN(Density-Based Spatial Clustering of Applications With Noise)**:具有噪声的基于密度的空间聚类方法

DBSCAN有两个重要的参数：

1. 邻域
2. 最小样本点数MinPts

假设样本集是，则：

邻域：对于，其邻域包含样本集中与的距离不大于的子样本集，即，这个子样本集的个数记为。

最小样本点数MinPts：对于，如果其邻域中的子样本集满足，则为核心对象，一个类中包含一个或多个核心对象。

密度直达：如果位于的邻域中，且为核心对象，则称由密度直达，注意反之不一定成立，除非也为核心对象，所以密度直达不具有对称性。

密度可达：对于和，如果存在样本序列，满足，，且由密度直达（这表明均为核心对象），则称由密度可达。注意，反之密度可达不一定成立，原因是密度直达不具有对称性。

密度相连：对于和，如果存在核心对象，使得和均由密度可达，则称和密度相连，

DBSCAN：由密度可达关系导出的最大密度相连的样本集合，即为最终聚类的一个类别，或者一个簇。

DBSCAN(DB, distFunc, eps, minPts) {

C = 0 */\* 类计数器 \*/*

**for each** point P **in** database DB {

**if** label(P) ≠ undefined **thencontinue***/\* Previously processed in inner loop \*/*

Neighbors N = RangeQuery(DB, distFunc, P, eps) */\* 寻找近邻样本 \*/*

**if** |N| < minPts **then** { */\* 密度检测 \*/*

label(P) = Noise */\* 噪声 \*/*

**continue**

}

C = C + 1 */\* next cluster label \*/*

label(P) = C */\* 标记为类C\*/*

Seed set S = N \ {P} */\* 近邻扩展，先剔除掉自己，然后寻找密度相连的所有样本， \*/*

**for each** point Q **in** S { */\* Process every seed point \*/*

**if** label(Q) = Noise **then** label(Q) = C */\* 将之前标记为噪声的标记为类C，视为边界点 \*/*

**if** label(Q) ≠ undefined **thencontinue***/\* Previously processed \*/*

label(Q) = C */\* Label neighbor \*/*

Neighbors N = RangeQuery(DB, distFunc, Q, eps) */\* 寻找Q的近邻 \*/*

**if** |N| ≥ minPts **then** { */\* Density check \*/*

S = S ∪ N */\* 如果Q也是核心对象，则将Q的近邻合并到P的近邻中，认为它们都是同一类（因为密度相连） \*/*

}

}

}

}

RangeQuery(DB, distFunc, Q, eps) {

Neighbors = empty list

**for each** point P **in** database DB { */\* Scan all points in the database \*/*

**if** distFunc(Q, P) ≤ eps **then** { */\* Compute distance and check epsilon \*/*

Neighbors = Neighbors ∪ {P} */\* Add to result \*/*

}

}

**return** Neighbors

}

从上述算法可以看出，DBSCAN是先找到一个核心对象，然后把与其密度相连的点都划分为一类。

DBSCAN的优点：

1. 可以对任意形状的稠密数据集进行聚类，K-Means和层次聚类一般用于凸数据集。
2. 不需要设定聚类数目
3. 对噪声和离群值不敏感
4. 聚类结果没有偏倚，K-Means受初始值影响较大

DBSCAN的缺点：

1. 运行速度较慢，最坏时间复杂度为。
2. 如果样本集的密度不均匀，聚类间距相差很大时，效果很差
3. 相同的参数无法保证传回相同的聚类，原因是如果数据集中边界点对多个类别都合适，会按照访问的顺序来确定，这是随机的。

**1.3 高斯混合模型(Gaussian Mixture Models):GMM**

GMM假设样本集中每一类样本都符合N维正态分布，即高斯分布。

假设样本集，这些样本是从k个高斯分布的数据里抽取的，将隐含的类别标签用表示，可以认为符合某种分布，其中。在给定后，服从多种高斯分布，即。由此可得：

利用极大似然估计函数：



两边去对数：



利用贝叶斯公式：

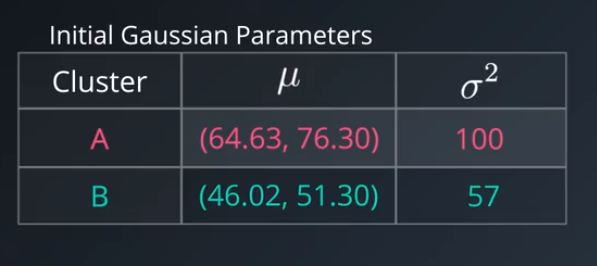


EM算法的具体推导过程参见D:\study\Python学习\Python机器学习\吴恩达机器学习讲义\机器学习课程个人学习笔记（上）\EM算法.pdf

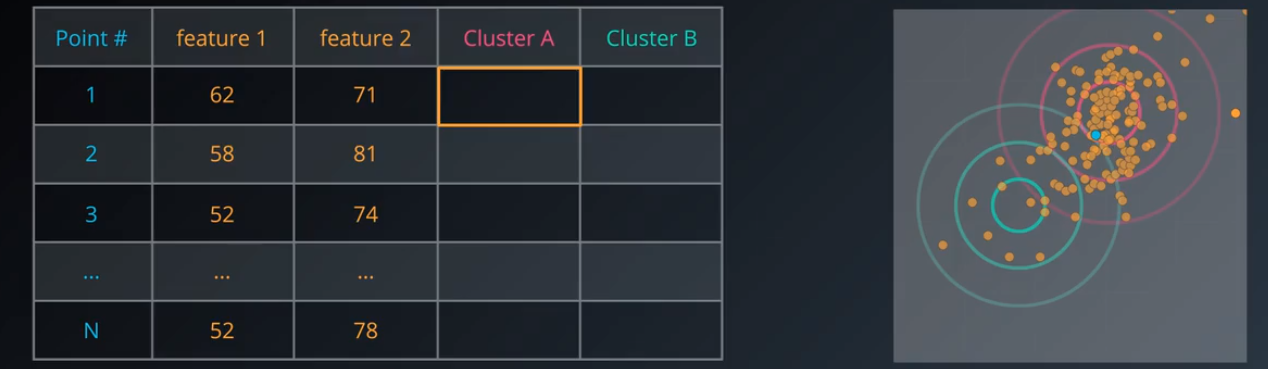
Expectation-Maximization For Gaussian Mixtures：EM期望最大算法

Udacity的介绍：以数据集包含两个高斯分布的类为例

1. 初始化k个高斯分布，本例中k=2。初始化高斯分布，需要均值和标准差，可以将其设置为数据集本身的平均值，也可以使用K-Means聚类，将聚类后的结果来初始化高斯分布，还可以像Udacity中的例子那样，赋值随机值。注意：初始化高斯参数很重要，会对EM结果产生重大影响。



1. E Step-期望步骤，对数据集进行软聚类。计算样本对每个类的隶属度。本例中包含两个类，A和B。



数据集有N个点。隶属度：



其中表示隐藏类别标签。

正态分布的概率密度函数：



计算隶属度（即分类概率）：把样本值代入到概率密度函数中



1. M-Step：期望最大化，重新估计高斯分布的参数。

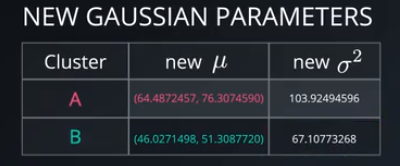






类B的更新与之类似。

更新完成后：



1. 评估对数似然



其中表示K所有分类数，表示与聚类相关的混合系数。

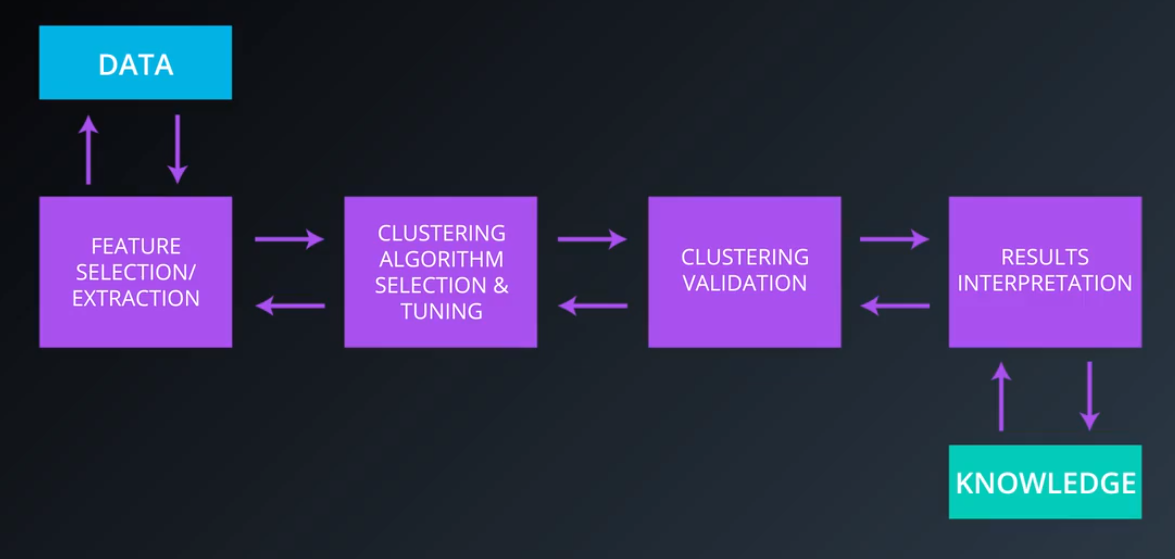
目标是使最大化，产生影响的参数有3个，，。

GMM的优点：

1. 给出样本属于每个类的隶属度，提供软聚类的信息
2. 聚类的形状复杂多样

GMM的缺点：

1. 对初始值敏感
2. 可能会收敛到局部最优值，并且收敛速度慢
3. 需要指定类别数目K
4. **聚类分析过程**



1. 特征选择与抽取
2. 聚类算法选择与提取，包括距离选择，如果欧氏距离，余弦距离等
3. 聚类校验
4. 结果评价

聚类校验指标：主要考虑紧凑性和可分性

* 1. **外部指标**

样本中有原始标签（非监督学习一般没有标签）。

<https://blog.csdn.net/sinat_30203515/article/details/82634778#%E5%85%B0%E5%BE%B7%E7%B3%BB%E6%95%B0rand-index>

调整兰德系数ARI：

先介绍兰德系数RI。

假设样本集合为，其中真实标签为，聚类结果为。

设定：

为在中为同一类且在中也为同一类的数据点对数。

为在中为同一类但在中隶属不同类别的数据点对数。

为在中不在同一类但在中为同一类的数据点对数。

为在中不在同一类且在中也不在同一类的数据点对数。



RI的值在[0,1]之间，当聚类完美匹配时，RI为1.

兰德系数在随机划分时，不是一个接近0的数值，从上述计算公式就可以看出，随机划分下，和很难都接近0.

调整兰德系数ARI：



表示同在真实类别和分类类别内的数据点数目，为类的数据点数目，为类的数目。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class\Cluster |  |  | … |  | Sums |
|  |  |  | … |  |  |
|  |  |  | … |  |  |
| … | … | … | … | … | … |
|  |  |  | … |  |  |
| Sums |  |  | … |  |  |



其中，相当于排列组合中的。参考D:\study\Python学习\Python机器学习\RI和ARI.pdf



调整兰德系数的优点：

1. 随机聚类的ARI都非常接近0
2. 取值在[-1, 1]之间，负数代表结果不好，越接近1越好
3. 可以用于聚类算法的比较

缺点：

需要真实标签

* 1. **内部指标**

不要求样本中有真实标签。

轮廓系数：可以用于查找最佳的聚类数目K。注：使用DBCAN时，不要使用轮廓系数，轮廓系数没有噪声的概念，计算每个样本的轮廓系数。

轮廓系数的取值在[-1, 1]之间，越接近1越好。



：样本i到簇内其它样本的平均距离，称为簇内不相似度，其值越小，说明样本i应该被聚类到该簇凑。簇内所有样本的均值为簇不相似度

样本i到其它簇样本的平均距离为，则，称为簇间不相似度，其值越大，说明样本i不属于其它簇。

接近1，说明样本i聚类合理

接近-1，则说明样本i需要被聚类到其它簇

接近0，说明样本i在两簇的边界。

聚类结果的轮廓系数为，表示聚类的好坏。

1. **特征缩放**

特征预处理的重要步骤。特征缩放的目标就是数据规范化使得特征的范围具有可比性，避免某些特征值过大，导致由它们主导识别而忽略其他的有用特征。另外使用梯度下降算法时，特征缩放可以加快收敛速度。

* 1. **调节比例**



* 1. **均值规范化**



* 1. **标准化**



* 1. **缩放到单位长度**



决策树和Logistic回归不会受特征缩放的影响，决策树在进行分类时，特征之间互不影响，判断一个特征也不会考虑使用另一个特征的数据，Logistic回归中，每个特征都有一个对应的特征系数，特征缩放，系数也会跟着变化，所以也不受特征缩放的影响。

SVM和K-Means聚类则会受到特征缩放的影响，原因是需要使用多个特征来计算距离。但凡需要使用多个特征计算距离的，都会受到特征缩放的影响。

1. **PCA(Principal components analysis)主成分分析**

特征降维。简单来说，数据集原本包含许多特征，PCA是通过找出（映射变换）几个综合特征（不再是原来特征中的数据）来替代原来的特征，这些综合特征尽可能包含原来所有特征的信息量，并且彼此之间互不相关。

PCA一般是通过将原来的特征进行线性组合，得到新的综合特征。选择线性组合时，希望其能尽可能的反应原来特征的信息，一般选取方差大的线性组合。

设样本集合为个样本个特征，假定：



其中

PCA将个特征转换为个综合特征。



上述模型满足以下条件：

1. 互不相关，即。
2. 
3. 

称为第一主成分，为第二主成分，以此类推。这里称为主成分系数。

PCA的计算步骤：

1. 数据标准化处理



标准化：



其中，

1. 计算样本相关系数矩阵



其中，

1. 求相关系数矩阵的特征值和对应的特征向量，其中特征值为主成分的方差，特征向量就是主成分系数。
2. 计算主成分的贡献率，并根据累计贡献率来选择主成分的个数。



1. 选择前个主成分特征向量，组成矩阵，然后将标准化后的原始数据来得到新的数据：



就是所谓的得分矩阵。

在sklearn中PCA得到的最大主成分个数为训练数据和特征数之间的较小值。

PCA的优点：

1. 用少数特征代替了多数特征，有效降低了计算量和特征筛选的难度。
2. 用方差来衡量信息，不受样本标签的限制
3. 各主成分正交，可以消除原始数据特征间的相互影响

PCA的缺点：

1. 新生成的特征含义模糊
2. 有时贡献率小的主成分可能会包含样本差异的重要信息。
3. PCA忽略类别标签，有可能会忽略重要的分类信息。
4. **随机投影（Random Projection）与ICA（Independent Component Correlation Analysis，独立成分分析）**

这也是用来降维的。

* 1. **随机投影**

主要用于维度很高的数据集，因为高维数据集如果使用PCA会非常繁琐并且消耗资源。

<https://blog.csdn.net/u012185296/article/details/78663914>

PCA与随机投影的简单比较



PCA会寻找方差最大的方向，而随机投影就是随机选择。

设原始数据集为，随机投影后的数据为：



随机投影可以在一定程度上保留样本之间的距离信息，所以它比较适用于依靠距离的算法。

<https://blog.csdn.net/MrCharles/article/details/80054441>

根据Jonhson-Lindenstrauss引理，设为原始样本点，，为投影后的样本点，，则：



其中eps表示一个可接受的误差系数。

**5.2 ICA**

ICA主要用于分离多个混合在一起的样本集，典型的如盲源分离。

典型的鸡尾酒宴会问题，假设party中有n个人，他们可以同时说话（每个人发出的声音信号是独立的），我们从n个麦克风得到m组数据组成的数据集，i表示时间序列。其中每组数据都是n维的，1个麦克风接收到1维数据，每1维都代表一个人的声音，每组数据都是n个人声音的混合。

ICA的假设：

1. 样本中每1维数据是独立采集的
2. 每个人的声音信号是非高斯分布。

样本集，每一个列向量是一个时刻n个麦克风采集的数据，目标对象集，目标对象集是n个对象，每个对象有m维数据。存在一个变换矩阵A，使得：





每个时刻采集的样本数据都是n个声音的线性组合。其中X为，A为，S为。不能是高斯分布。

尽管没有明确的说明，但应该认为。

令，则：



ICA算法的目的就是得到最佳的W。

Udacity中介绍的FastICA算法：

1. 中心化+漂白

中心化：，中心化后，样本的均值就为0.

漂白：将乘以一个矩阵变成，使得的协方差为单位矩阵，即：



令，利用特征值分解EVD，得到特征向量E和特征值对角矩阵D，则：



1. 选择初始随机矩阵W
2. 更新W



其中为非二次函数，可以选择：，等。

1. 去相关操作，
2. 如果不收敛，调到3)继续。