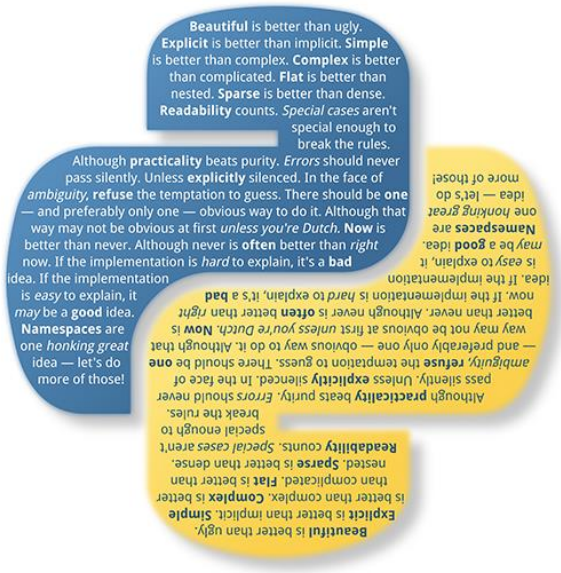
CDA 数据分析师
CERTIFIED DATA ANALYST

CDA大数据分析师就业班
之
Python机器学习



python™

聚类算法

美国总统大选

2000年和2004年的美国总统大选，候选人的得票数非常接近。如果有1%的选民将手中的选票投向另外的候选人，那么选举结果就会截然不同。实际上，如果妥善加以引导和吸引，少部分的选民就会转换立场，而这些人的立场对选举结果将产生非常大的影响。

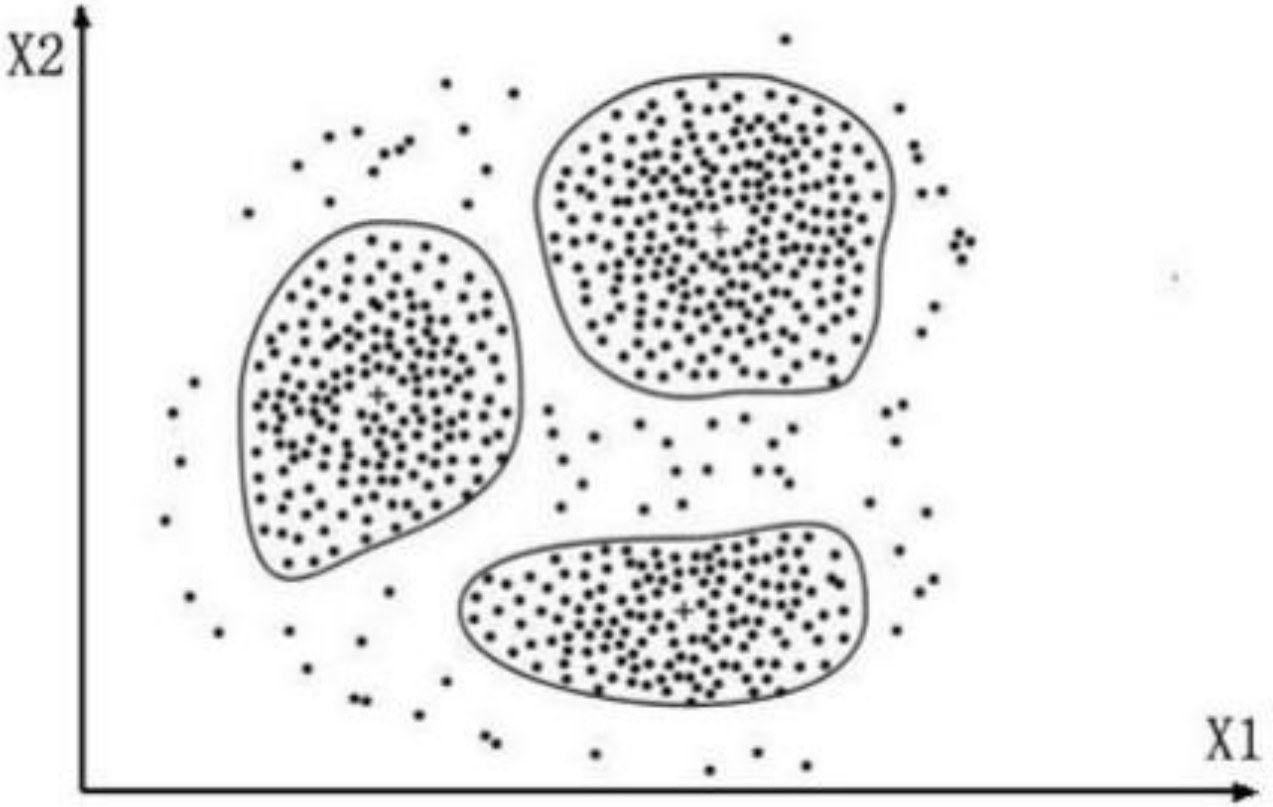
美国总统大选

http://finance.sina.com.cn/roll/2017-02-07/doc-ifyafcyx7385795.shtml?qq-pf-to=pcqq.group

http://www.sohu.com/a/126377260_116132

聚类

聚类跟分类有什么区别？



寻找优质客户

二八定律无处不在

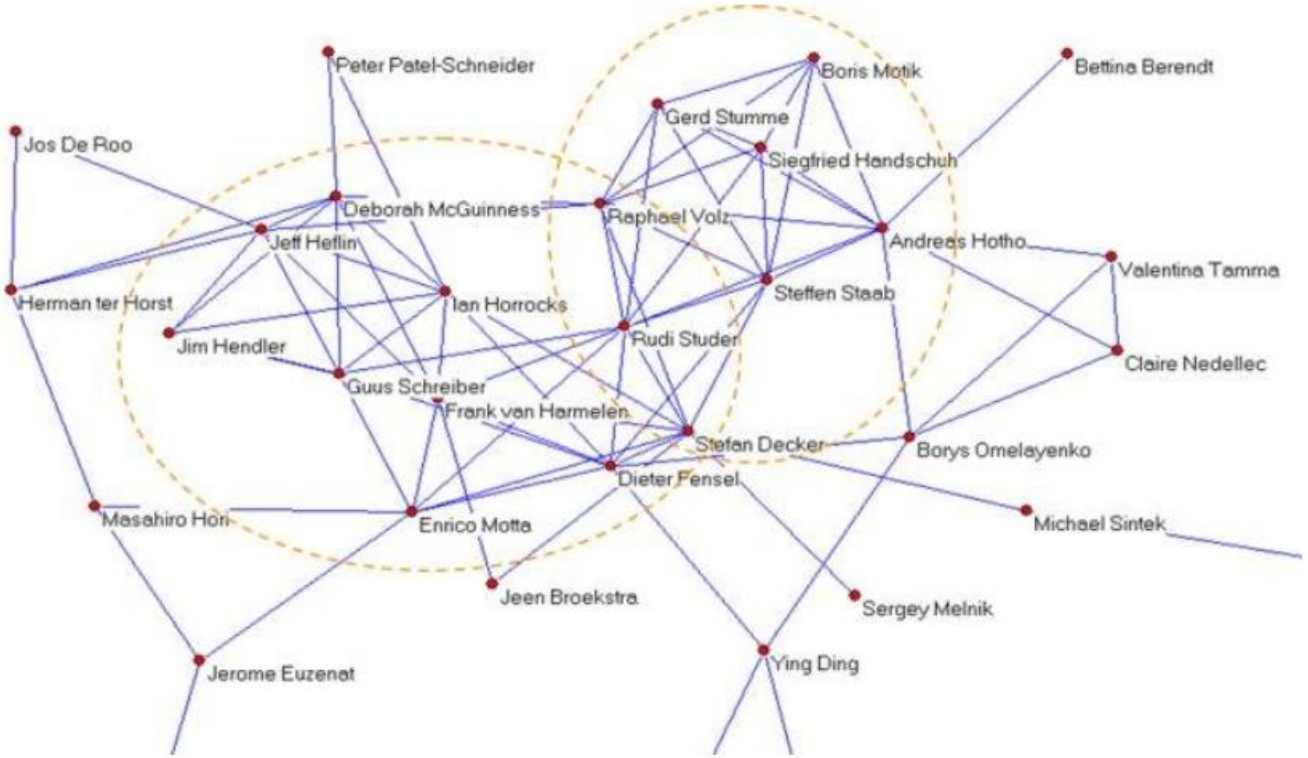
20%的用户提供了银行80%的利润来源

20%的用户消费了运营商话费总额的80%

公司中20%的员工完成了80%的工作

社会中20%的人拥有80%的话语权

社区发现


Peter Patel-Schneider
Boris Motik
Bettina Berendt
Gerd Stumme
Jos De Roo
Siegfried Handschuh
Deborah McGuinness
Raphael Volz
Jeff Heflin
Andreas Hotho
Valentina Tamma
Herman ter Horst
Ian Horrocks
Steffen Staab
Rudi Studer
Jim Hendler
Claire Nedellec
Guus Schreiber
Frank van Harmelen
Stefan Decker
Borys Omelayenko
Dieter Fensel
Masahiro Hori
Enrico Motta
Michael Sintek
Jeen Broekstra
Sergey Melnik
Jerome Euzenat
Ying Ding

异常点监控

信用卡诈骗

黑客攻击

```
xmenu=1&inajax=1" "Mozilla/4.0 (compatible: MSIE 7.0: Windows NT 5.1: .NET CLR 1.1.4322: .NET CLR 2.0.50727: .NET CLR 3.0.045
06.30)"
183.3.51.76 - - [29/Nov/2013:01:27:25 +0800] "GET /member.php?mod=logging&action=login HTTP/1.1" 200 17707 "http://r.dataguru
.cn/member.php?mod=logging&action=login" "Mozilla/4.0 (compatible: MSIE 7.0: Windows NT 5.1: .NET CLR 1.1.4322: .NET CLR 2.0.
50727: .NET CLR 3.0.04506.30)"
183.3.51.76 - - [29/Nov/2013:01:27:26 +0800] "GET /member.php?mod=logging&action=login HTTP/1.1" 200 17707 "http://r.dataguru
.cn/member.php?mod=logging&action=login" "Mozilla/4.0 (compatible: MSIE 7.0: Windows NT 5.1: .NET CLR 1.1.4322: .NET CLR 2.0.
50727: .NET CLR 3.0.04506.30)"
183.3.51.76 - - [29/Nov/2013:01:27:26 +0800] "POST /member.php?mod=logging&action=login&loginsubmit=yes&inajax=1&ajaxmenu=1 H
TTP/1.1" 200 297 "http://r.dataguru.cn/member.php?mod=logging&action=login&loginsubmit=yes&inajax=1&ajaxmenu=1" "Mozilla/4.0
(compatible: MSIE 7.0: Windows NT 5.1: .NET CLR 1.1.4322: .NET CLR 2.0.50727: .NET CLR 3.0.04506.30)"
66.249.64.1 - - [29/Nov/2013:01:30:19 +0800] "GET /home.php?mod=space&uid=50144&do=home&view=me&from=space HTTP/1.1" 200 5769
 "-" "Mozilla/5.0 (iPhone: CPU iPhone OS 6_0 like Mac OS X) AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0 Mobile/10A5376
e Safari/8536.25 (compatible: Googlebot-Mobile/2.1: +http://www.google.com/bot.html)"
66.249.64.8 - - [29/Nov/2013:01:30:44 +0800] "GET /space-uid-73446.html HTTP/1.1" 200 4782 "-" "Mozilla/5.0 (compatible: Goog
lebot/2.1: +http://www.google.com/bot.html)"
210.51.177.136 - - [29/Nov/2013:01:35:28 +0800] "GET / HTTP/1.0" 200 46531 "-" "User-Agent: Mozilla/5.0 (compatible: MSIE 6.0
;Windows XP)"
66.249.64.1 - - [29/Nov/2013:01:36:52 +0800] "GET /space-uid-73384.html HTTP/1.1" 200 4776 "-" "Mozilla/5.0 (compatible: Goog
lebot/2.1: +http://www.google.com/bot.html)"
66.249.64.1 - - [29/Nov/2013:01:38:25 +0800] "GET /space-uid-73345.html HTTP/1.1" 200 4434 "-" "Mozilla/5.0 (compatible: Goog
lebot/2.1: +http://www.google.com/bot.html)"
183.3.20.129 - - [29/Nov/2013:01:38:45 +0800] "GET /member.php?mod=logging&action=login HTTP/1.1" 200 17707 "http://r.datagur
u.cn/member.php?mod=logging&action=login" "Mozilla/4.0 (compatible: MSIE 7.0: Windows NT 5.1: .NET CLR 1.1.4322: .NET CLR 2.0
.50727: .NET CLR 3.0.04506.30)"
183.3.20.129 - - [29/Nov/2013:01:38:49 +0800] "GET /member.php?mod=logging&action=login HTTP/1.1" 200 17707 "http://r.datagur
u.cn/member.php?mod=logging&action=login" "Mozilla/4.0 (compatible: MSIE 7.0: Windows NT 5.1: .NET CLR 1.1.4322: .NET CLR 2.0
.50727: .NET CLR 3.0.04506.30)"
183.3.20.129 - - [29/Nov/2013:01:38:49 +0800] "POST /member.php?mod=logging&action=login&loginsubmit=yes&inajax=1&ajaxmenu=1
HTTP/1.1" 200 297 "http://r.dataguru.cn/member.php?mod=logging&action=login&loginsubmit=yes&inajax=1&ajaxmenu=1" "Mozilla/4.0
 (compatible: MSIE 7.0: Windows NT 5.1: .NET CLR 1.1.4322: .NET CLR 2.0.50727: .NET CLR 3.0.04506.30)"
[root@class2room web_logs]#
```

天文学研究


Image credit: NASA/JPL-Caltech/E. Churchwell (Univ. of Wisconsin, Madison)

K-MEANS

K-MEANS

- 算法接受参数 k ；然后将事先输入的n个数据对象划分为 k个聚类以便使得所获得的聚类满足：同一聚类中的对象相似度较高；而不同聚类中的对象相似度较小。

- 算法思想：以空间中k个点为中心进行聚类，对最靠近他们的对象归类。通过迭代的方法，逐次更新各聚类中心的值，直至得到最好的聚类结果

K-MEANS

1.先从没有标签的元素集合A中随机取k个元素，作为k个子集各自的重心。
2.分别计算剩下的元素到k个子集重心的距离（这里的距离也可以使用欧氏距离），根据距离将这些元素分别划归到最近的子集。
3.根据聚类结果，重新计算重心（重心的计算方法是计算子集中所有元素各个维度的算数平均数）。
4.将集合A中全部元素按照新的重心然后再重新聚类。
5.重复第4步，直到聚类结果不再发生变化。

非连续型数据

表 8.4: 项目、类目和样本的取值情况

样本	性别		外语				专业			职业	
	男	女	英	日	德	俄	统计	会计	金融	教师	工程师
$x_{(1)}$	1	0	1	0	0	0	0	0	1	0	1
$x_{(2)}$	0	1	1	0	0	0	1	0	0	1	0

记 m_1 为 $x_{(i)}$ 和 $x_{(j)}$ 在 m 个项目所有类目中 $1-1$ 配对的总数，m_0 为 $0-0$ 配对的总数，m_2 为不配对的总数. 显然，有

$$m_0 + m_1 + m_2 = p.$$

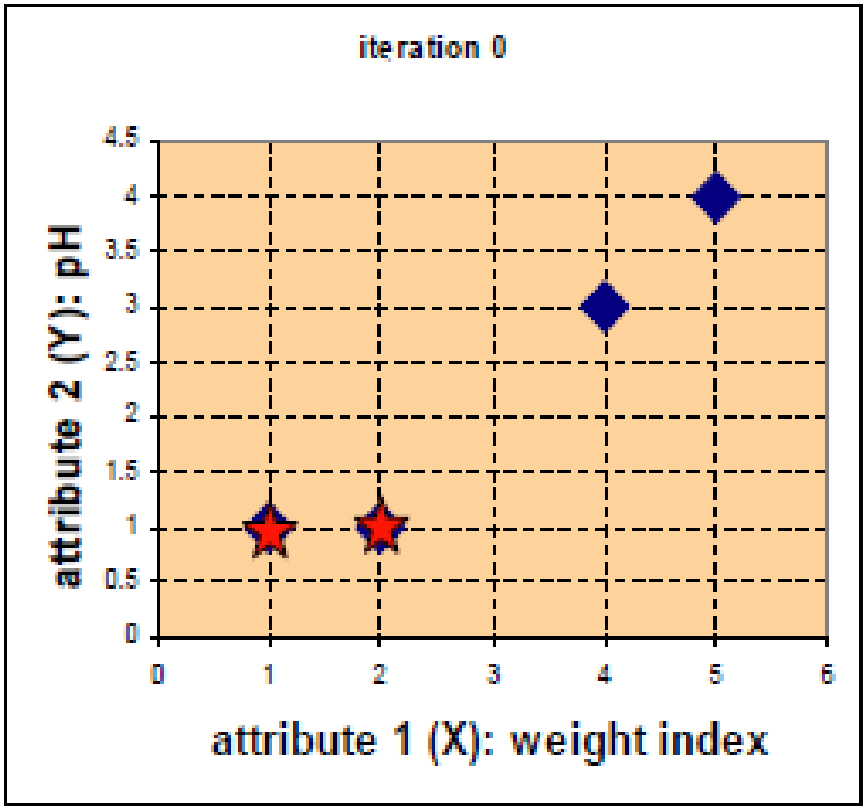
样本 $x_{(i)}$ 和 $x_{(j)}$ 之间的距离可以定义为

$$d_{ij} = \frac{m_2}{m_1 + m_2}. \tag{8.63}$$

对于表 8.4 中的数据，$m_0 = 4, m_1 = 1, m_2 = 6$. 因此，距离为 $d_{12} = 6/7 = 0.8571429$.

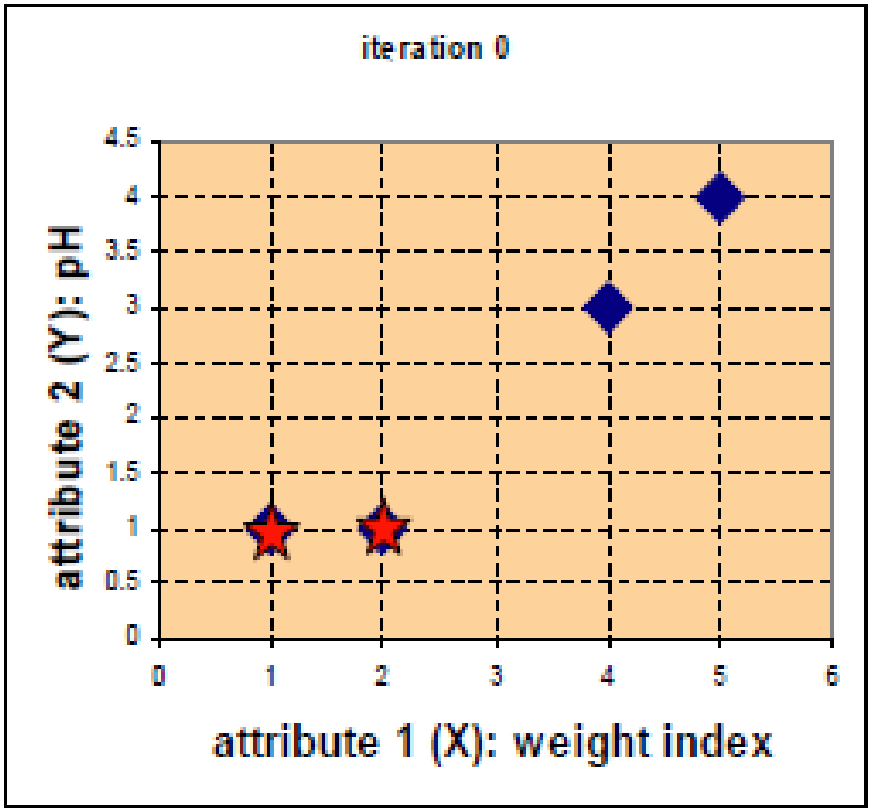
例子

Object	Feature 1 (X): weight index	Feature 2 (Y): pH
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4



假设取(1,1)(2,1)为两个分类中心点

例子



$$\mathbf{D}^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \quad \begin{matrix} c_1 = (1,1) & group-1 \\ c_2 = (2,1) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

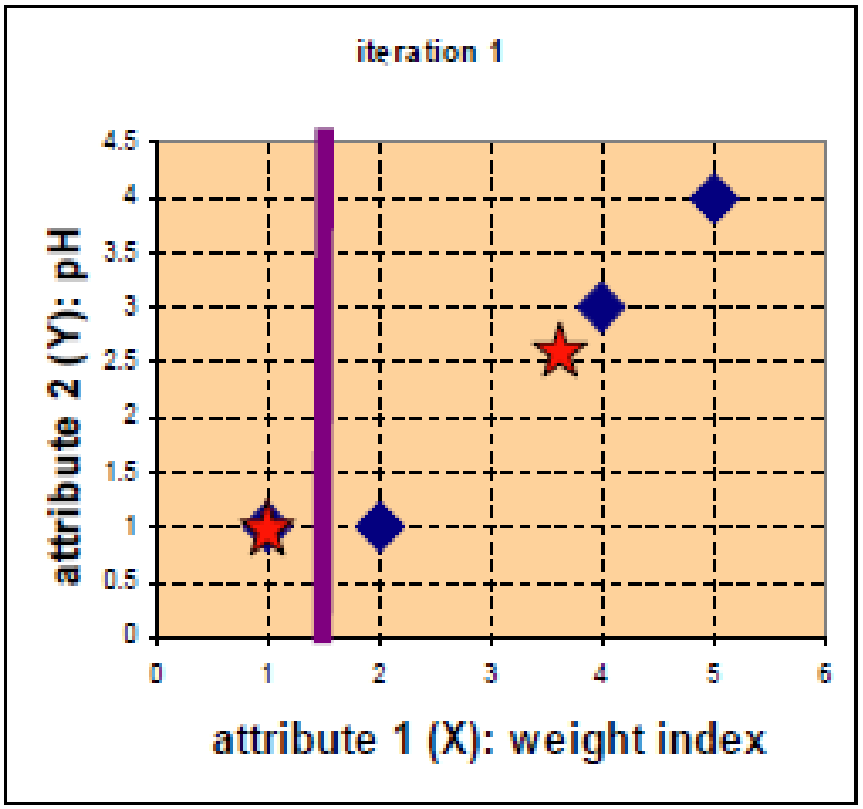
$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

$$\mathbf{G}^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$c_2 = \left(\frac{2+4+5}{3}, \frac{1+3+4}{3}\right) = \left(\frac{11}{3}, \frac{8}{3}\right)$$

例子



$$\mathbf{D}^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (\frac{11}{3}, \frac{8}{3}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

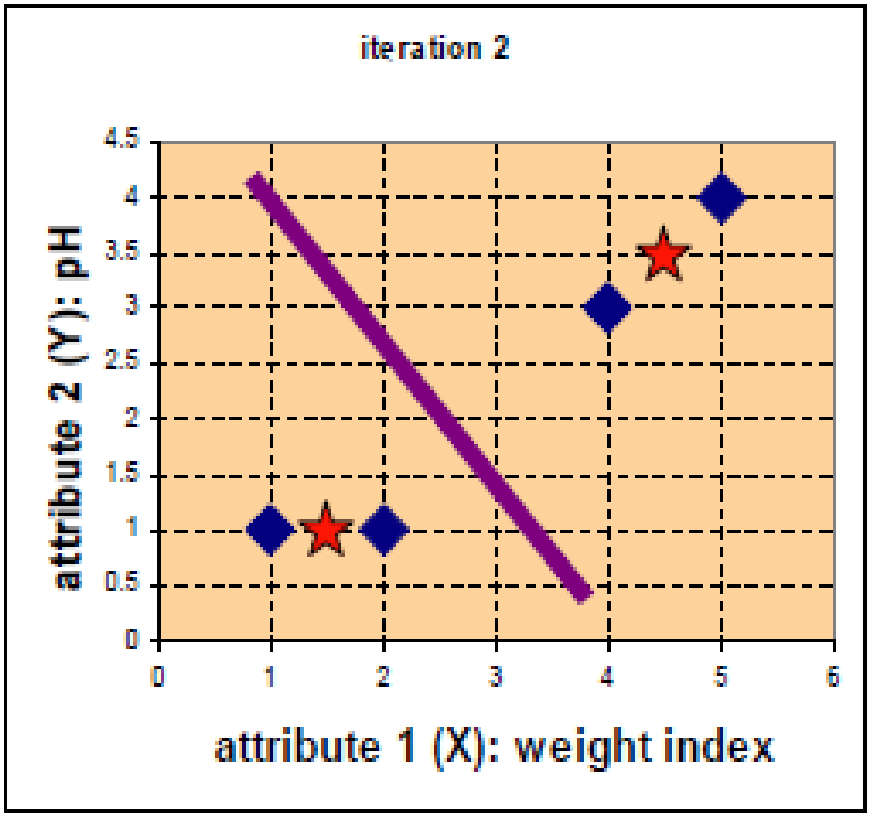
$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$

$$\mathbf{G}^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\mathbf{c}_1 = (\frac{1+2}{2}, \frac{1+1}{2}) = (1\frac{1}{2}, 1) \text{ and } \mathbf{c}_2 = (\frac{4+5}{2}, \frac{3+4}{2}) = (4\frac{1}{2}, 3\frac{1}{2})$$

例子



$$D^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} c_1 = (1\frac{1}{2}, 1) & group-1 \\ c_2 = (4\frac{1}{2}, 3\frac{1}{2}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

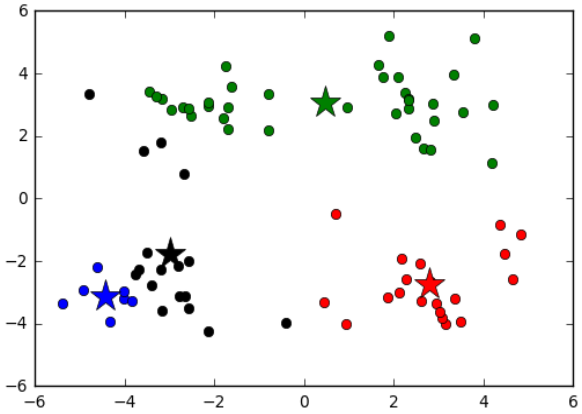
$$G^2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

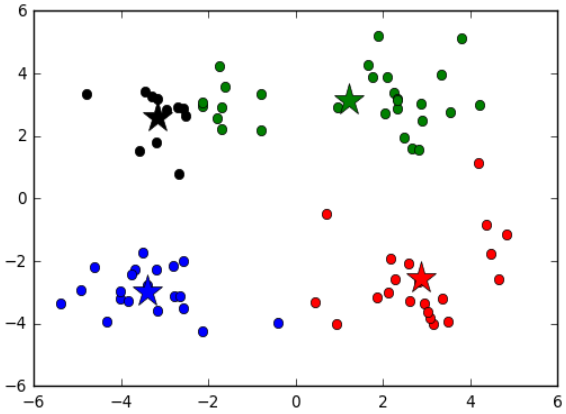
聚类不发生变化，算法迭代停止

K-means算法

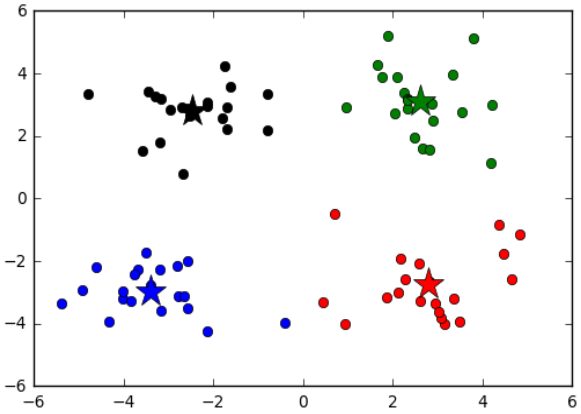
迭代了1次



迭代了5次



迭代了9次



python实现K-MEANS

Talk is cheap

Show me the

CODE

sklearn-K-MEANS

Talk is cheap

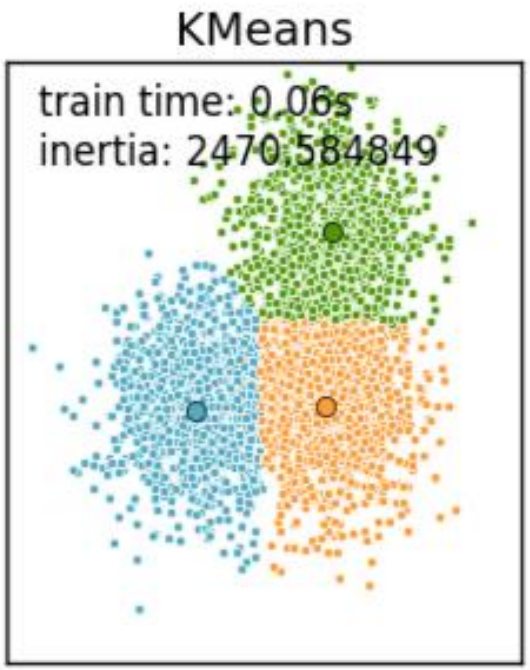
Show me the

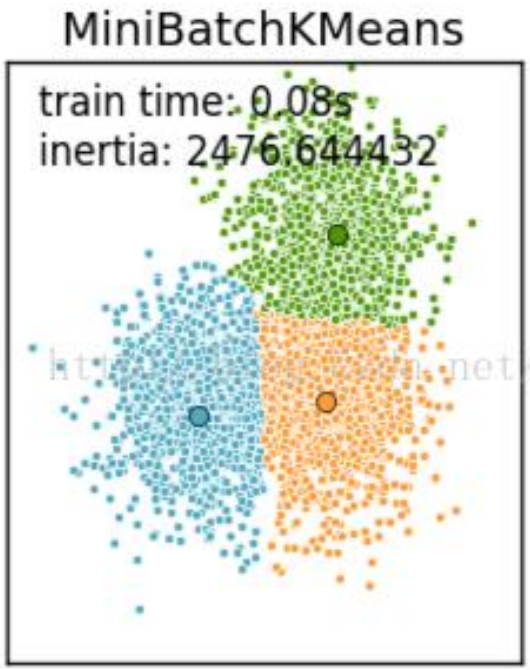
CODE

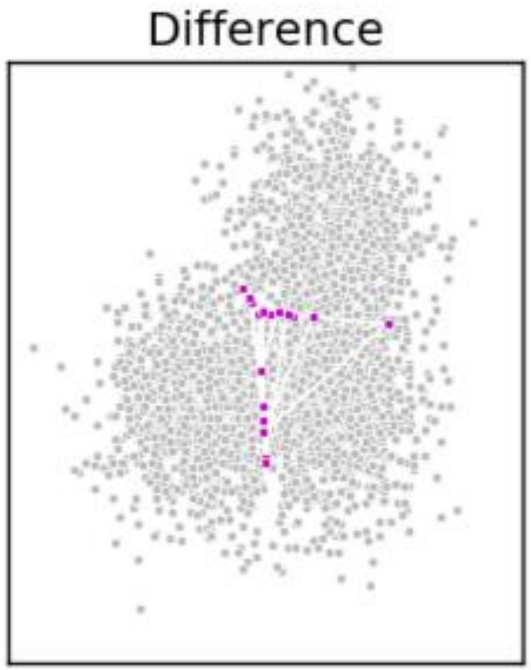
Mini Batch K-Means

Mini Batch K-Means算法是K-Means算法的变种，采用小批量的数据子集减小计算时间。这里所谓的小批量是指每次训练算法时所随机抽取的数据子集，采用这些随机产生的子集进行训练算法，大大减小了计算时间，结果一般只略差于标准算法。
该算法的迭代步骤有两步：
1：从数据集中随机抽取一些数据形成小批量，把他们分配给最近的质心
2：更新质心
与K均值算法相比，数据的更新是在每一个小的样本集上。
Mini Batch K-Means比K-Means有更快的 收敛速度，但同时也降低了聚类的效果，但是在实际项目中却表现得不明显。

Mini Batch K-Means







sklearn-Mini-Batch-K-MEANS

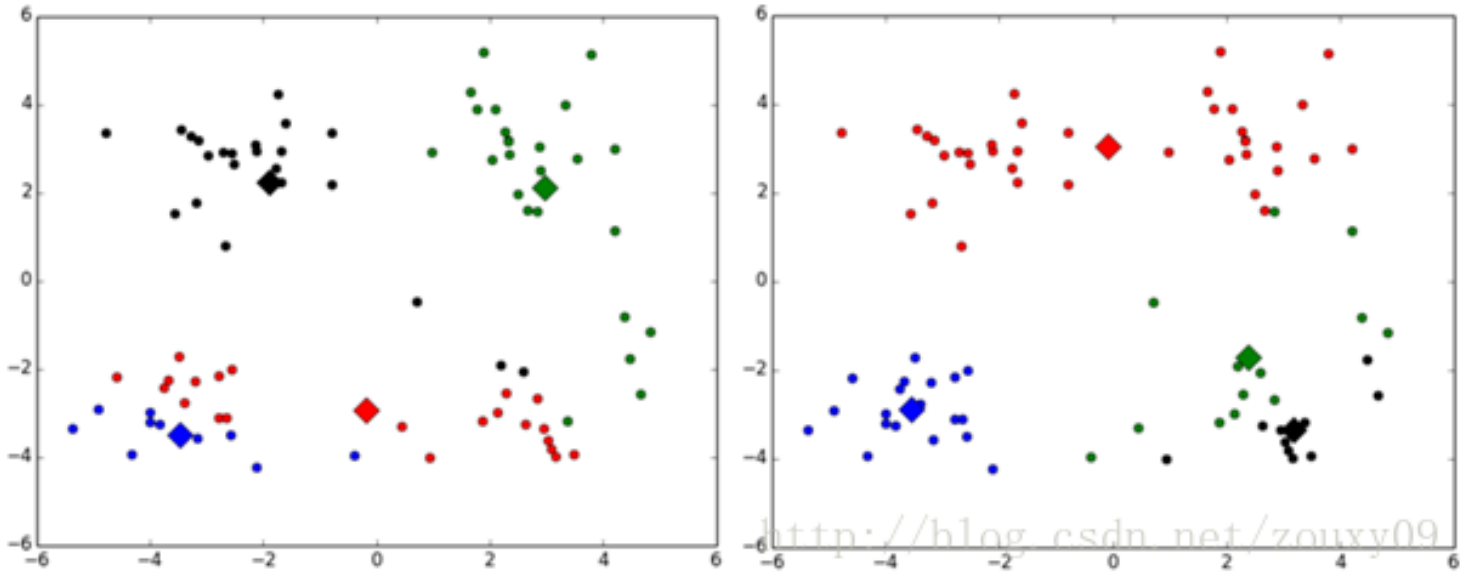
Talk is cheap

Show me the

CODE

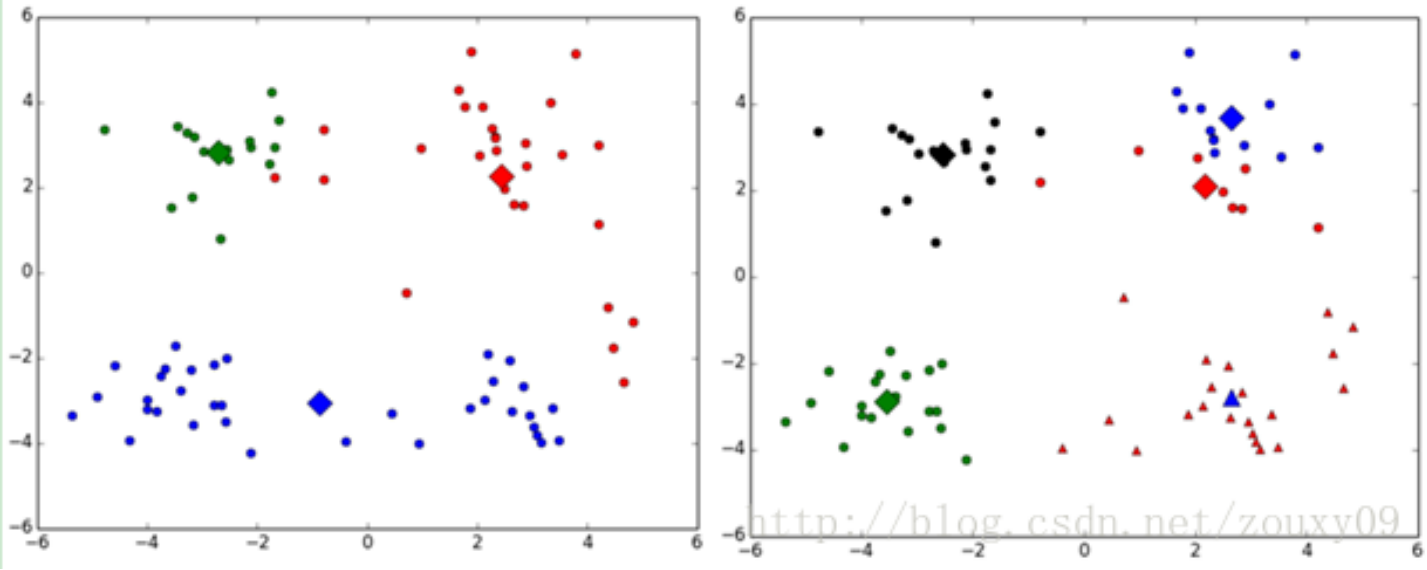
算法分析1

对k个初始质心的选择比较敏感，容易陷入局部最小值。例如，我们上面的算法运行的时候，有可能会得到不同的结果，如下面这两种情况。K-means也是收敛了，只是收敛到了局部最小值：



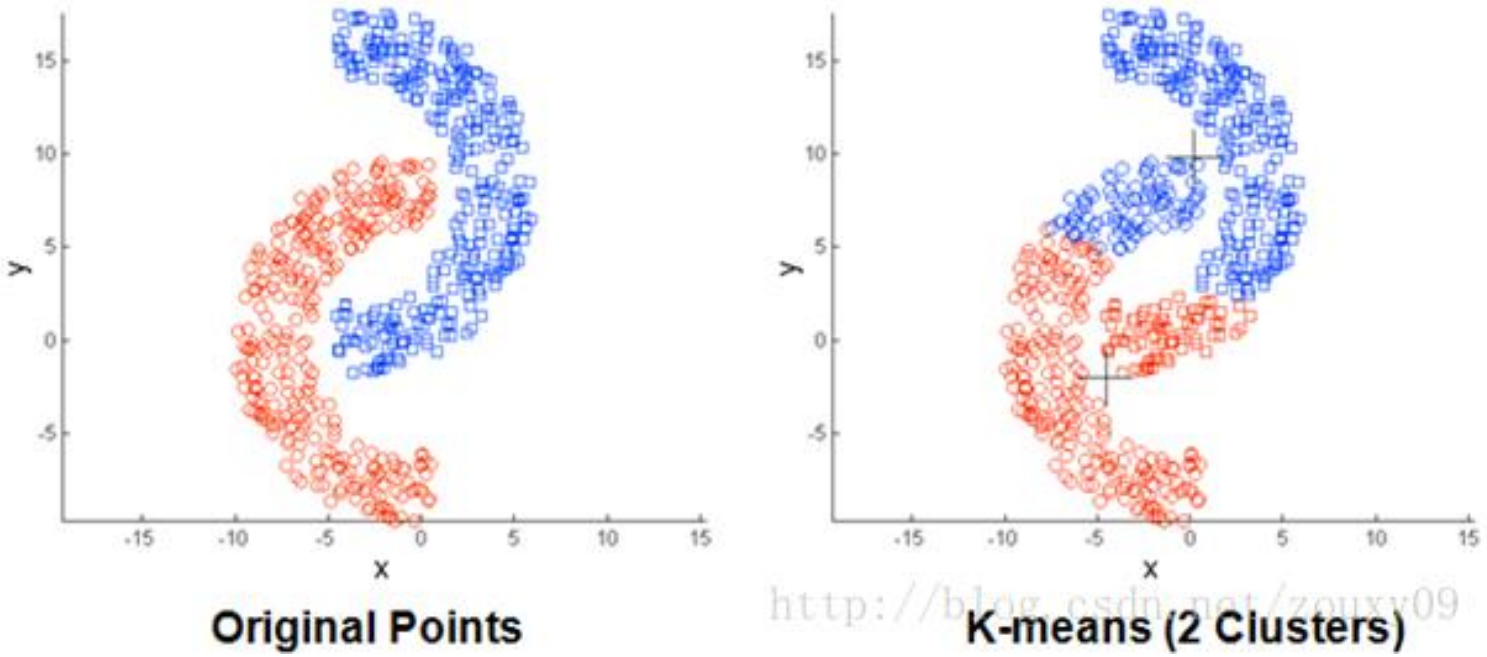
算法分析2

k值的选择是用户指定的，不同的k得到的结果会有挺大的不同，如下图所示，左边是k=3的结果，蓝色的簇太稀疏了，蓝色的簇应该可以再划分成两个簇。右边是k=5的结果，红色和蓝色的簇应该合并为一个簇。



算法分析3

存在局限性，如下面这种非球状的数据分布就搞不定了：



算法分析4

数据库比较大的时候，收敛会比较慢。

K-Means算法优化1

使用多次的随机初始化，计算每一次建模得到的代价函数的值，选取代价函数最小结果作为聚类结果。

```
For i = 1 to 100 {

      Randomly initialize K-means.
      Run K-means. Get
.
      Compute cost function (distortion)

      }
```

$$J(c^{(1)},\ldots,c^{(m)},\mu_1,\ldots,\mu_K)=\frac{1}{m}\sum_{i=1}^{m}||x^{(i)}-\mu_{c^{(i)}}||^2$$

python实现K-MEANS优化1

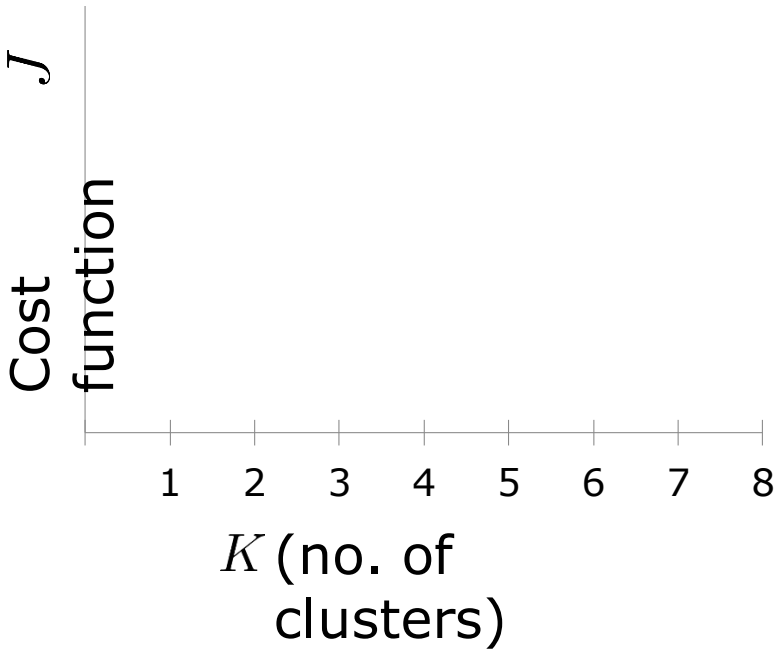
Talk is cheap

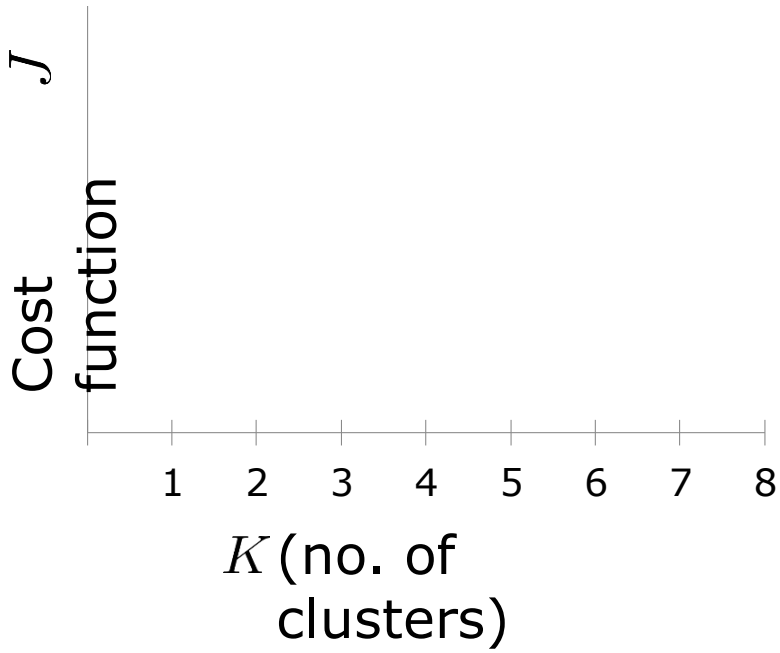
Show me the

CODE

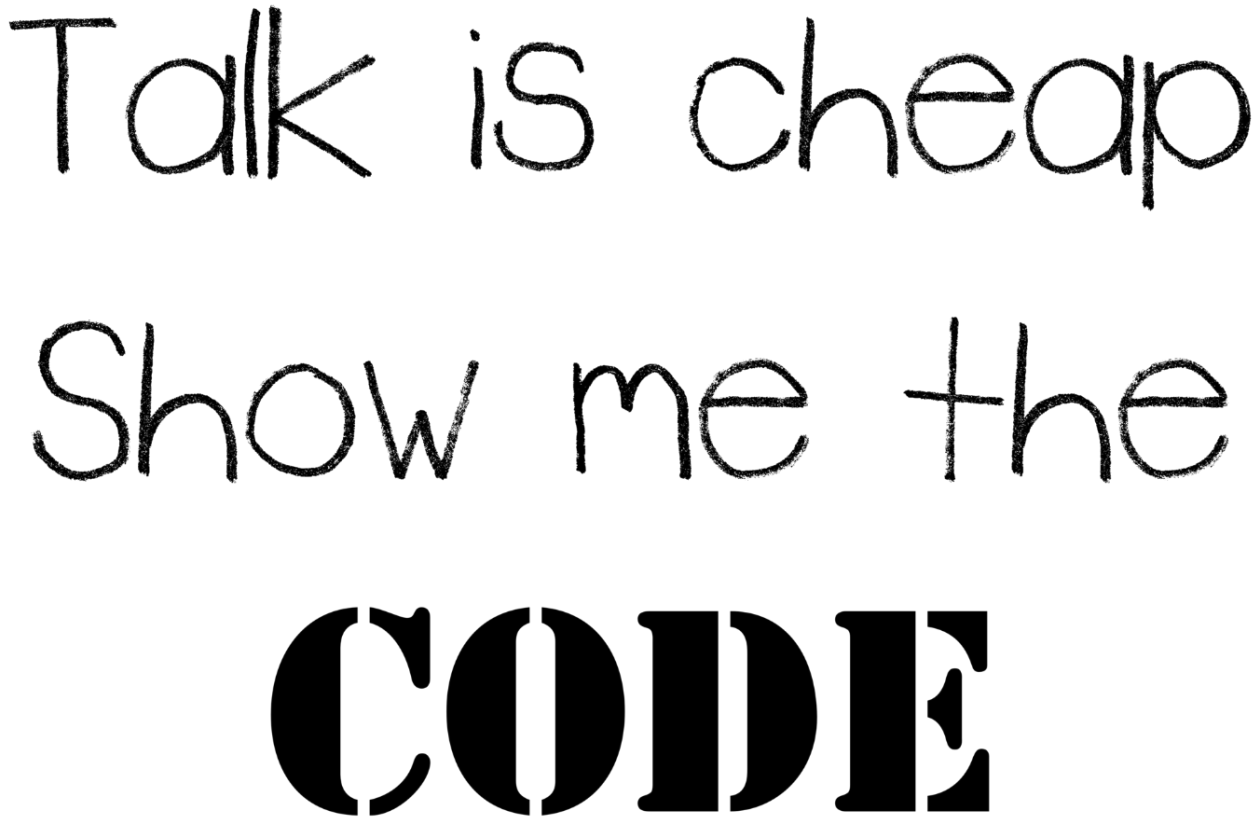
K-Means算法优化2

使用肘部法则来选择k的值：





python实现K-MEANS优化2



可视化K-MEANS

https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

K-MEANS球员聚类练习

篮球运动员数据。完整数据集包括5个特征，每分钟助攻数、运动员身高、运动员出场时间、运动员年龄和每分钟得分数。聚类判断球员是什么位置（控位、分位、中锋等）

	assists_per_minute	height	time_played	age	points_per_minute
0	0.0888	201	36.02	28	0.5885
1	0.1399	198	39.32	30	0.8291
2	0.0747	198	38.80	26	0.4974
3	0.0983	191	40.71	30	0.5772
4	0.1276	196	38.40	28	0.5703
5	0.1671	201	34.10	31	0.5835
6	0.1906	193	36.20	30	0.5276
7	0.1061	191	36.75	27	0.5523
8	0.2446	185	38.43	29	0.4007
9	0.1670	203	33.54	24	0.4770
10	0.2485	188	35.01	27	0.4313

K-MEANS球员聚类练习

Talk is cheap

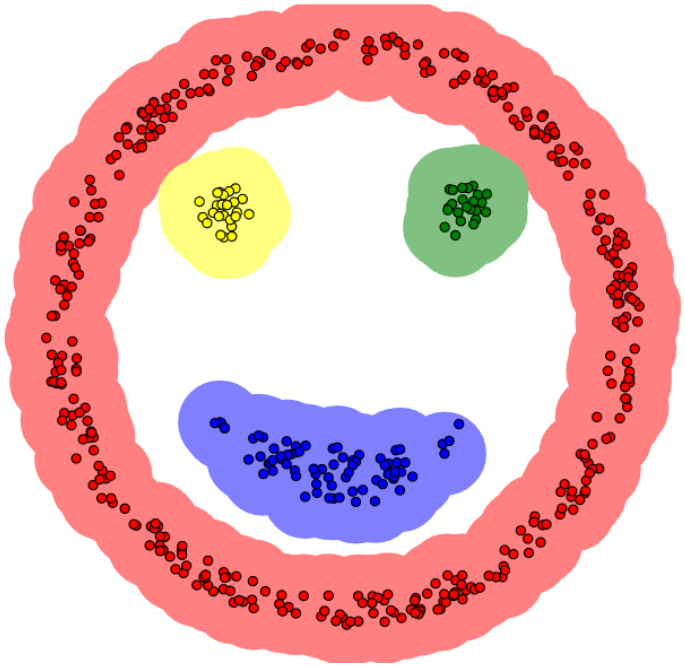
Show me the

CODE

DBSCAN

基于密度的方法：DBSCAN

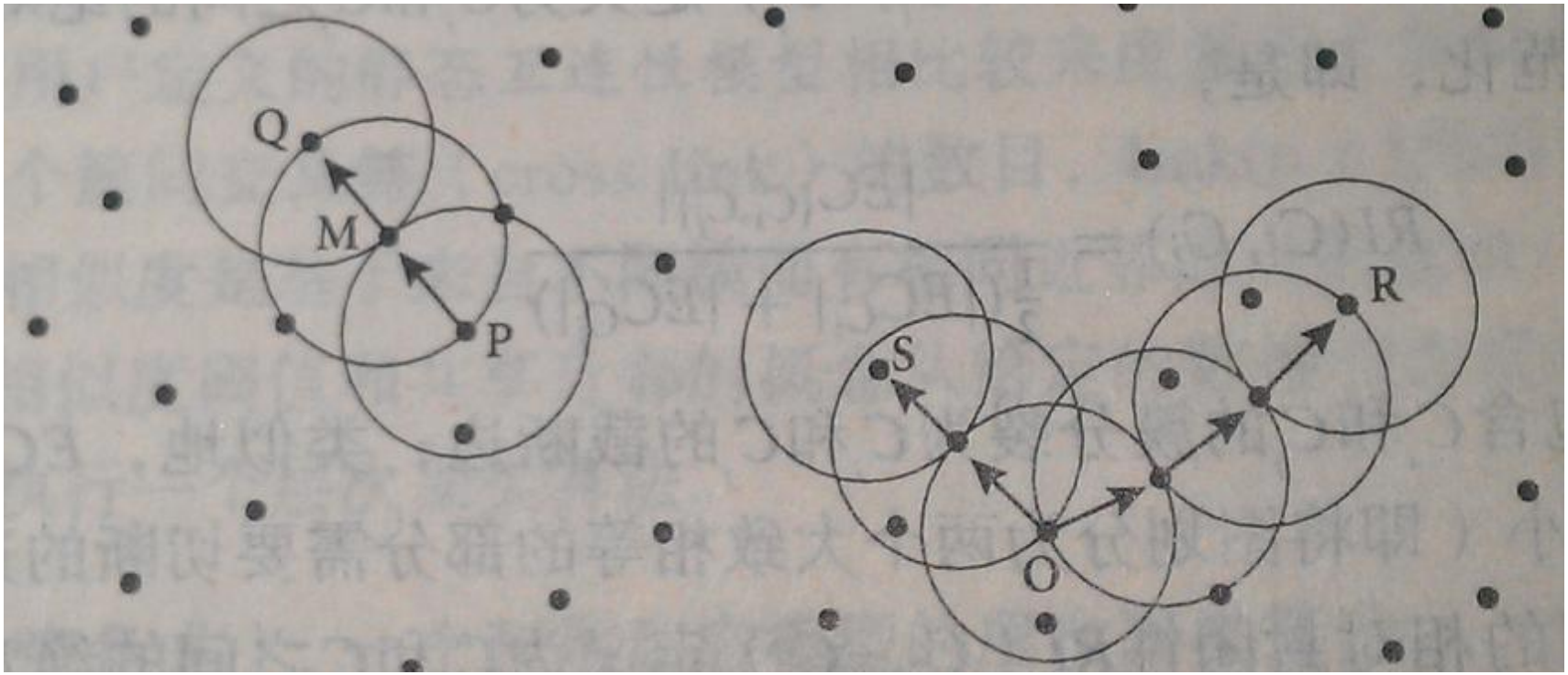
DBSCAN = Density-Based Spatial Clustering of Applications with Noise
本算法将具有足够高密度的区域划分为簇，并可以发现任何形状的聚类



DBSCAN

ε邻域：给定对象半径ε内的区域称为该对象的ε邻域。
核心对象：如果给定 ε 邻域内的样本点数大于等于Minpoints，则该对象为核心对象。
直接密度可达：给定一个对象集合D，如果p在q的ε邻域内，且q是一个核心对象，则我们说对象p从q触发是直接密度可达的(directly density-reachable)。
密度可达：集合D，存在一个对象链p1,p2…pn,p1=q,pn=p,pi+1是从pi关于ε和Minpoints直接密度可达，则称点p是从q关于ε和Minpoints密度可达的。
密度相连：集合D存在点o，使得点p、q是从o关于ε和Minpoints密度可达的，那么点p、q是关于ε和Minpoints密度相连的。

DBSCAN



DBSCAN算法思想

1.指定合适的ε和Minpoints。
2.计算所有的样本点，如果点p的ε邻域里有超过Minpoints个点，则创建一个以p为核心点的新族。
3.反复寻找这些核心点直接密度可达（之后可能是密度可达）的点，将其加入到相应的簇，对于核心点发生“密度相连”状况的簇，给予合并。
4.当没有新的点可以被添加到任何簇时，算法结束。

DBSCAN分析

缺点：

- 当数据量增大时，要求较大的内存支持I/O消耗也很大。
- 当空间聚类的密度不均匀、聚类间距差相差很大时，聚类质量较差。

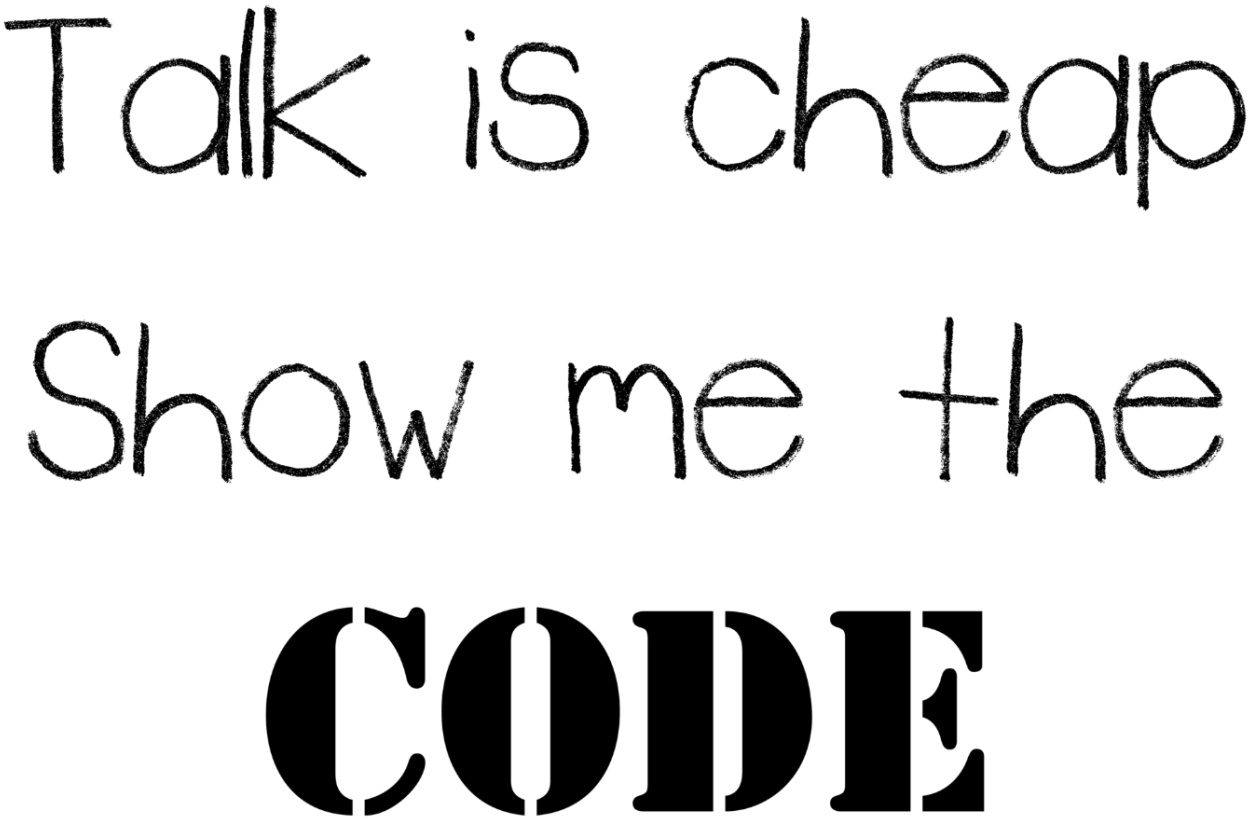
DBSCAN和K-MEANS比较：

- DBSCAN不需要输入聚类个数。
- 聚类簇的形状没有要求。
- 可以在需要时输入过滤噪声的参数。

DBSCAN可视化

https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

sklearn-DBSCAN1



sklearn-DBSCAN2(练习)