KNN & K-means
以及模型的评价

统计与数学学院

杨晓蓉

Part Ⅰ　KNN算法简介

- 全称：K-Nearest Neighbor
- 简称：K-NN
- 中文：K-近邻算法

黑暗餐厅

顾客：在一个完全黑暗的餐厅里接受服务
服务员：仅凭触觉和听觉记忆在路上小心地移动

魅力所在：去掉一个人的视觉感官输入会增强他的味觉和嗅觉，从而以一种全新并且感到兴奋的方式来体验食物

数据收集

感受最突出最特别的香味和口感是什么？
食物尝起来是甜还是咸？

数据分析

将品尝到的一小口与之前的体验进行对比，得出结论

如果该食物闻起来像只鸭子，并且品尝起来也像只鸭子，那么你很可能是在吃鸭子

K-NN算法是怎么来的

思想：相似的东西可能具有相似的属性。利用这个原理，可以对数据进行分类，将其划分到最相近的类别或者最接近的邻居。

定义：为了判定未知样本的类别，以全部训练样本作为代表点，计算未知样本与所有训练样本的距离，并以最近邻者的类别作为决策未知样本类别的唯一依据。

适用的分类任务类型：分类特征和目标类之间的关系多种、复杂，用其他方式极难理解，但是具有相似类的项目又非常接近。即：一个概念很难定义，但是当你看到它时你知道它是什么，那么KNN就是最合适的分类方式。

一个简单的例子

想一想：下面图片中只有三种豆，有三个豆是未知的种类，如何判定他们的种类？

未知的豆离哪种豆最近就认为未知豆和该豆是同一种类

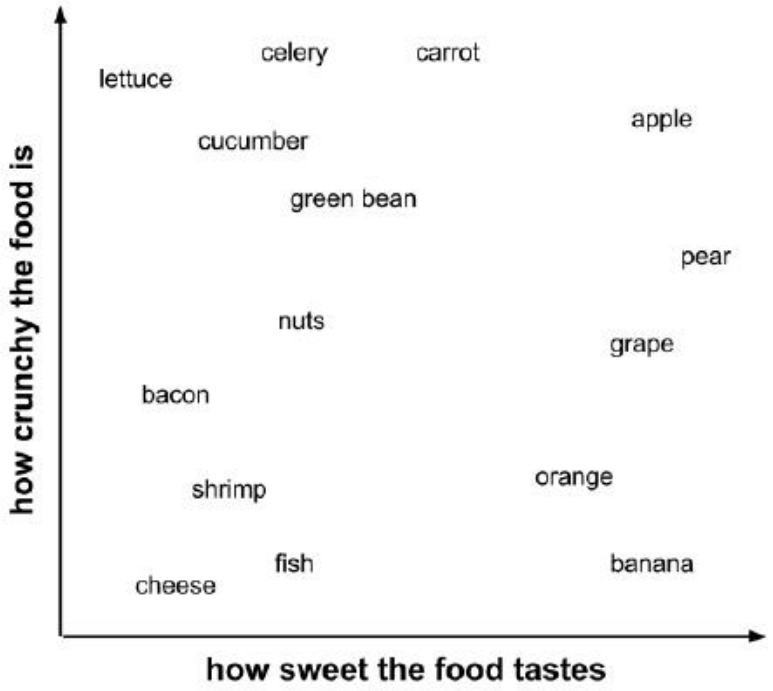
回到黑暗餐厅的例子

品味数据集合：假设记录每种食物配料（ingredient）的两个特征：脆度（crunchiness）1-10和甜度（sweetness）1-10。并根据食物的特征标记它们为下面3种类型之一：水果（fruit）、蔬菜（vegetable）或者蛋白质（protein）。

例如，下面的部分数据录入形式为：

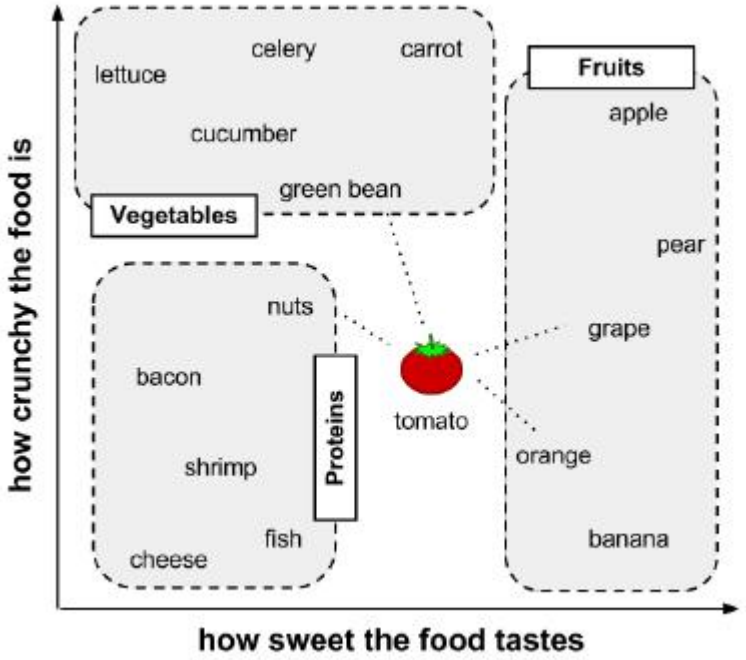
ingredient	sweetness	crunchiness	food
apple	10	9	fruit
bacon	1	4	protein
banana	10	1	fruit
carrot	7	10	vegetable
celery	3	10	vegetable
cheese	1	1	protein

多维数据的可视化：两个特征：crunchiness、sweetness在二维平面设上展示如下：



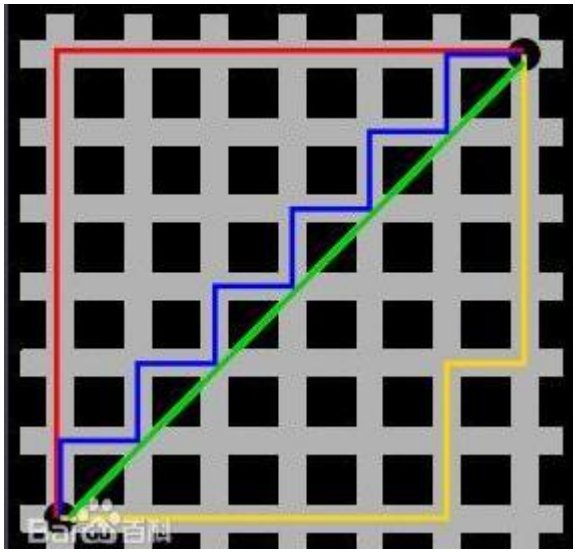
注：相似的食物趋向于聚集得更近

解决问题：西红柿（crunchiness=4；sweetness=6）是什么？蔬菜？还是水果？为什么？



要点1：计算距离——定位西红柿和其他案个案的距离或者相似性

常用的两种距离：点 $P(x_1, y_1)$ 和 $Q(x_2, y_2)$



- 欧式距离（**Euclidean distance**）

$$\text{dist}(P,Q) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

- 曼哈顿距离（**Manhattan distance**）

$$\text{dist}(P,Q) = | x_1 - x_2 | + | y_1 - y_2 |$$

欧式距离通过“直线距离”（crow flies）来度量，即最短的直接路线。另一种常见的距离度量是**曼哈顿距离**（Manhattan distance），该距离基于一个行人在城市街区步行所采取的路线。如果你有兴趣了解更多关于距离度量的方法，可以使用 ?dist 命令，阅读 R 中的距离函数文档（该文档本身就是一个很有用的工具）。

以欧式距离为例，西红柿（crunchiness=4；sweetness=6）和绿豆（crunchiness=7；sweetness=3）的距离为：

$$\text{dist}(\text{tomato, green bean}) = \sqrt{(6-3)^2 + (4-7)^2} = 4.2$$

以此类推，西红柿和其他几个近邻之间的距离如下表所示：

ingredient	sweetness	crunchiness	food type	distance to the tomato
grape	8	5	fruit	sqrt((6 − 8)^2 + (4 − 5)^2) = 2.2
green bean	3	7	vegetable	sqrt((6 − 3)^2 + (4 − 7)^2) = 4.2
nuts	3	6	protein	sqrt((6 − 3)^2 + (4 − 6)^2) = 3.6
orange	7	3	fruit	sqrt((6 − 7)^2 + (4 − 3)^2) = 1.4

- **K=1 (1NN)：orange是最近的，西红柿属于水果**
- **K=3 (3NN)：orange、grape和nut是最近的，少数服从多数，西红柿属于水果**

要点2：选择一个合适的K

- K的选择将决定把模型推广到未来数据时模型的好坏
- 偏差与方差的权衡(bias-variance tradeoff):

——选择一个大的K会减少噪声对模型的影响

——选择一个小的K则容易受异常值的影响，造成分类错误

考虑两种极端的情况：

1) K取所有样本个数——则变成少数服从多数的投票了，不管近邻是什么，最后的分类取决于占大多数的那个分类；

2)K取1——一旦最近邻被贴错了标签，而其他近邻才是正确的，样本就会本分到错误的类里面

- 实际应用中，K的选择取决于要学习概念的难度和训练数据中的案例数量，通常K=3~10，或者可以取样本总数的平方根。

要点3：数据的准备

- **KNN在应用之前，通常要讲特征转换到一个标准的范围内。**

假设在黑暗餐厅的数据中，再增加一个特征：

辛辣度(spiciness)——使用史高维尔指标(Scoville scale)来测量，取值1-100,0000不等。

特征取值调整的方法：

- **min-max标准化(min-max normalization)**

$$X_{new} = \frac{X - \min(X)}{\max(X) - \min(X)}$$

- **z-score标准化(z-score standardization)**

$$X_{new} = \frac{X - \mu}{\sigma} = \frac{X - \text{Mean}(X)}{\text{StdDev}(X)}$$

对于名义数据可采用哑变量编码，如：

- **性别(gender)：** $\text{gender} = \begin{cases} 1, & \text{male} \\ 0, & \text{female} \end{cases}$

- **温度(temperature, 分hot, medium, cold三种情况)**

$$\text{hot} = \begin{cases} 1, & \text{if hot} \\ 0, & \text{else} \end{cases}, \qquad \text{medium} = \begin{cases} 1, & \text{if medium} \\ 0, & \text{else} \end{cases}$$

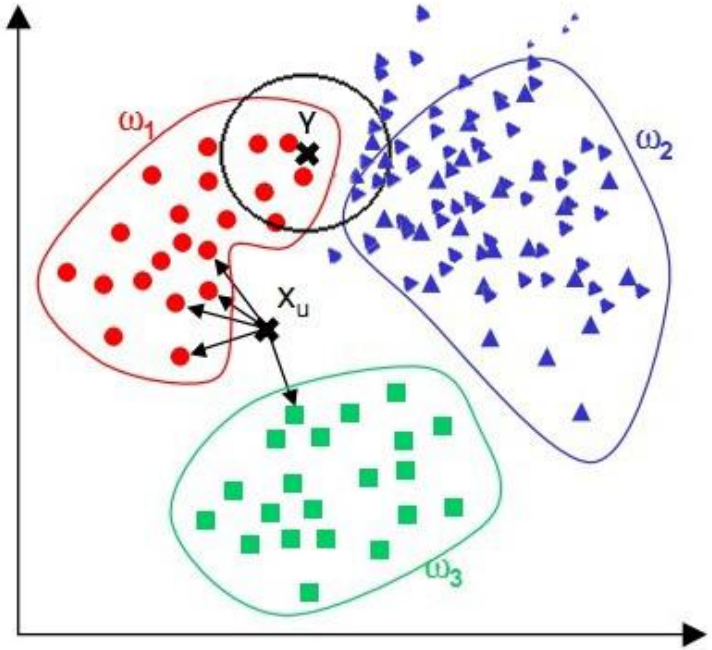
如果名义特征是有序的（可以将我们刚刚看到的温度变量作为例子），那么一种哑变量编码的替代方法就是给类别编号并且应用 min-max 标准化。例如，cold、warm 和 hot 可以编号为 1、2 和 3，min-max 标准化后为 0、0.5 和 1。使用该方法要注意的是，只有当你确信类别之间的步长相等时，才能应用该方法。例如，你可以证明，尽管 poor、middle class 和 wealthy 是有序的，但是 poor 和 middle class 之间的差异比 middle class 和 wealthy 之间的差异大（或小）。在这种情况下，哑变量编码是一种更保险的方法。

K-NN算法的优点和缺点

- K-NN算法本身简单有效，它是一种 lazy-learning 算法，分类器不需要使用训练集进行训练，训练时间复杂度为0。K-NN 分类的计算复杂度和训练集中的样本数目成正比。
- 近邻法的一个严重问题是需要存储全部训练样本，以及繁重的距离计算量。

优点	缺点
• 简单且有效 • 对数据的分布没有要求 • 训练阶段很快	• 不产生模型，在发现特征之间关系上的能力有限 • 分类阶段很慢 • 需要大量的内存 • 名义变量（特征）和缺失数据需要额外处理

K-NN算法不足举例



对于位置样本X_u，通过K-NN算法，我们显然可以得到X应属于红点，但对于位置样本Y，通过KNN算法我们似乎得到了Y应属于蓝点的结论，而这个结论直观来看并没有说服力。

用KNN算法诊断乳腺癌

Step1：数据收集

数据来源：UCI机器学习数据仓库

http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29

UCI

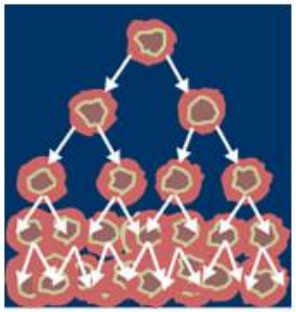
Machine Learning Repository

Center for Machine Learning and Intelligent Systems

Breast Cancer Wisconsin (Diagnostic) Data Set

Download: Data Folder, Data Set Description

Abstract: Diagnostic Wisconsin Breast Cancer Database



Data Set Characteristics:	Multivariate	**Number of Instances:**	569	**Area:**	Life
Attribute Characteristics:	Real	**Number of Attributes:**	32	**Date Donated**	1995-11-01
Associated Tasks:	Classification	**Missing Values?**	No	**Number of Web Hits:**	1288219

Step2：探索和准备数据

数据包括了569例细胞活检案例，每个案例有32个特征，第一列是ID，第二列是癌症诊断结果（B是良性的，M是恶性的），其他30个特征是细胞核的10个不同特征的均值、标准差和最大值。

- Radius
- Texture
- Perimeter
- Area
- Smoothness
- Compactness
- Concavity
- Concave points
- Symmetry
- Fractal dimension

每个特征记录他们的均值、标准差和最大值，构成了30维的特征向量

Step2：探索和准备数据

- ID需要删除
- 目标属性变量需要编码成因子型变量
- 数据中B有357例，M有212例
- 不同特征的取值范围差异很大，需要标准化

```
> summary(wbcd[c("radius_mean", "area_mean", "smoothness_mean")])
  radius_mean      area_mean      smoothness_mean
 Min.   : 6.981   Min.   : 143.5   Min.   :0.05263
 1st Qu.:11.700   1st Qu.: 420.3   1st Qu.:0.08637
 Median :13.370   Median : 551.1   Median :0.09587
 Mean   :14.127   Mean   : 654.9   Mean   :0.09636
 3rd Qu.:15.780   3rd Qu.: 782.7   3rd Qu.:0.10530
 Max.   :28.110   Max.   :2501.0   Max.   :0.16340
```

- 对数据进行标准化以后，创建训练数据集和测试数据集
- 前469条作为训练数据集，后100条作为测试集

当构造训练数据集和测试数据集时，保证每一个数据集都是数据全集的一个有代表性的子集是很重要的。在刚刚看到的案例中，记录已经按照随机顺序排列，所以我们可以简单地提取100个连续的记录来创建一个测试数据集。如果数据是按照非随机的模式排列的，比如按时间顺序或者以具有相似值的组的顺序，那么这将不是合适的创建上述两个数据集的方法。在这些情况下，需要用到随机抽样方程。

Step3：基于数据训练模型

- 利用KNN的包，导入预处理的数据进行分类判断，几个关键参数有：训练集、测试集、训练数据的分类标签（需用因子型变量）、K的取值

kNN 分类语法

应用 class 添加包中的函数 knn()

创建分类器并进行预测：

```
p <- knn(train, test, class, k)
```

- train：一个包含数值型训练数据的数据框
- test：一个包含数值型测试数据的数据框
- class：包含训练数据每一行分类的一个因子向量
- k：标识最近邻数目的一个整数

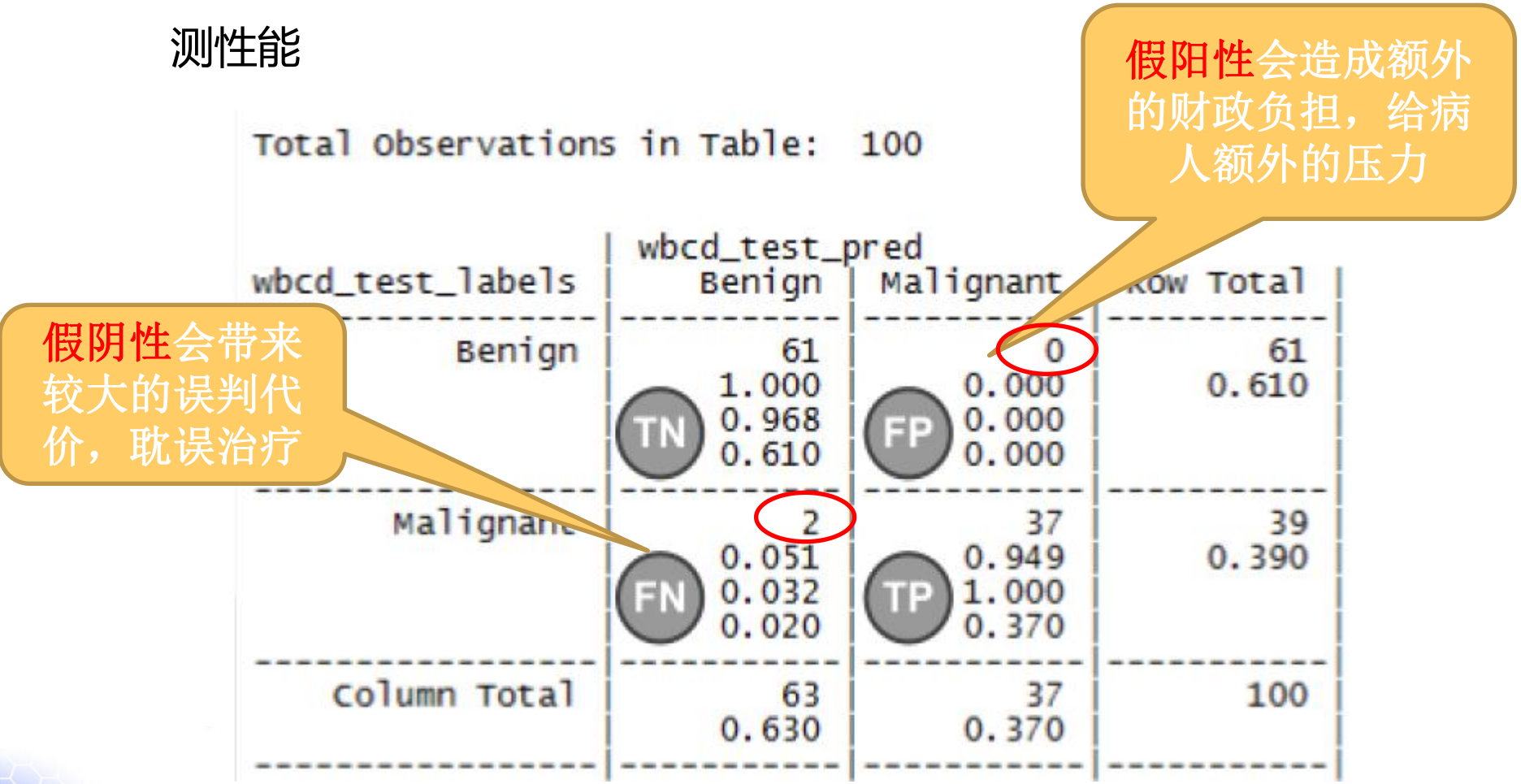
该函数返回一个因子向量，该向量含有测试数据框中每一行的预测分类。

例子：

```
wbcd_pred <- knn(train = wbcd_train, test = wbcd_test,)
                 cl =wbcd_train_labels, k = 3
```

Step4：评估模型的性能

- 通过测试集的预测分类和真实分类的比较结果，来衡量模型的预测性能



Step5：模型性能的提高

- 尝试转化z-score值

min-max会把数据压缩到0-1上，但是数据如果存在特殊情况，用min-max方法会给数据预定义最小和最大值，未必是个好的方法。而z-score则不会将数据压缩到0-1范围内。

wbcd_test_labels	wbcd_test_pred Benign	Malignant	Row Total
Benign	61 1.000 0.924 0.610	0 0.000 0.000 0.000	61 0.610
Malignant	5 0.128 0.076 0.050	34 0.872 1.000 0.340	39 0.390
Column Total	66 0.660	34 0.340	100

正确率并没有提升，反而下降了

Step5：模型性能的提高

- 测试其他K值

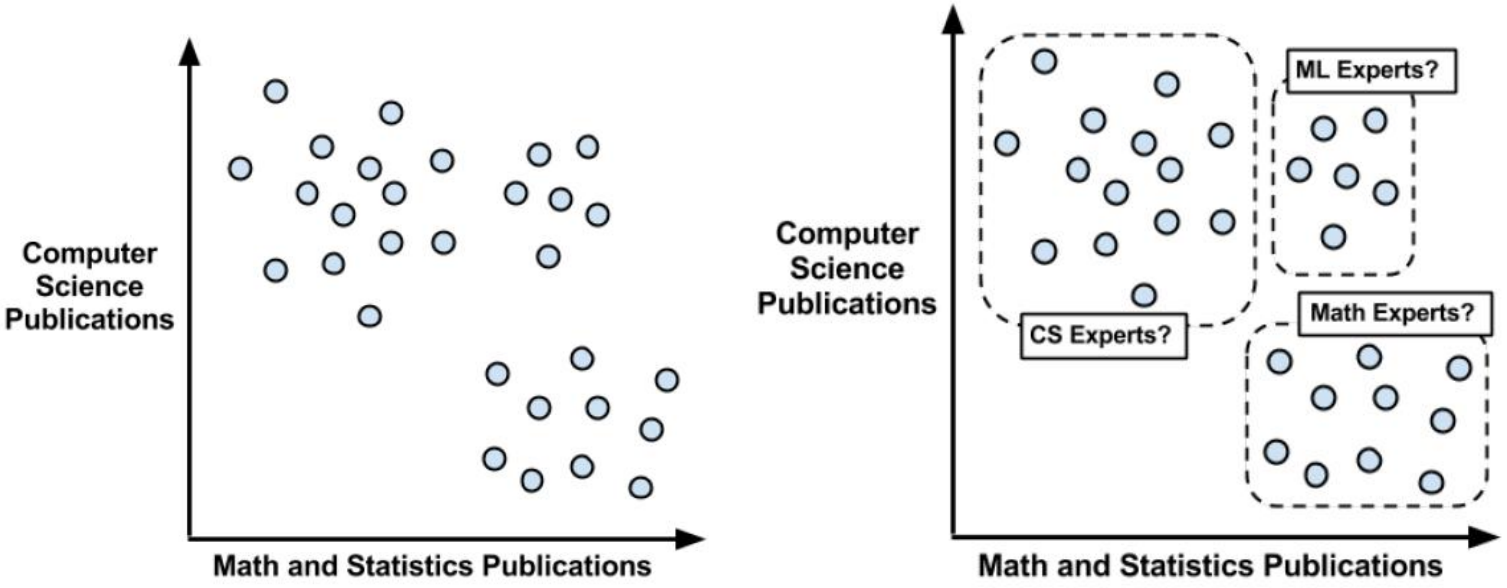
调试不同的K值，对相同的测试集进行测试，平衡假阴性和假阳性之间的数量。

k 值	假阴性的数量	假阳性的数量	错误分类的比例
1	1	3	4%
5	2	0	2%
11	3	0	3%
15	3	0	3%
21	2	0	2%
27	4	0	4%

Part II　K-means算法简介

- 全称：k-means
- 中文：K-均值聚类算法

- 假设在一场以数据科学为主题的学术会议上，你打算把参加会议的学这个根据他们的研究特长分三类：计算机与数据库专家、数学与统计专家、机器学习专家

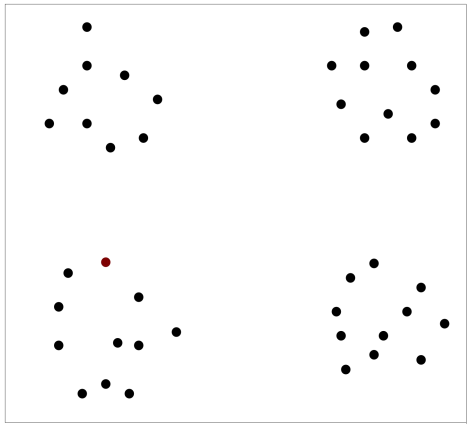


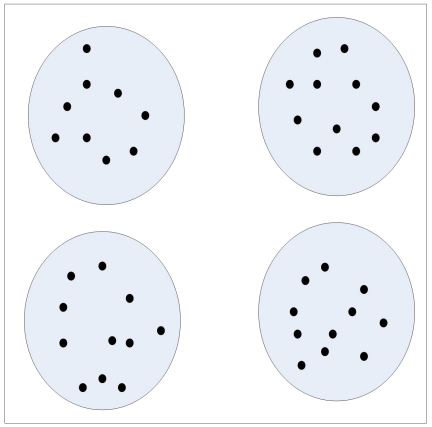
考察与会者的论文发表情况

- 这是一个无标签的分类，我们不知道每个点的真实值。通过聚类我们建立的分类标签。
- 可以接着从分类标签出发，再结合其他有监督的学习算法来寻找这些类中的重要的预测指标，这也成为了一个半监督学习**(semi-supervised learning)**

聚类

- 聚类（Clustering）就是对大量未知标注的数据集，按数据的内在相似性将数据集划分为多个族（Cluster），使族内的数据相似度尽可能大而类别间的数据相似度尽可能小。

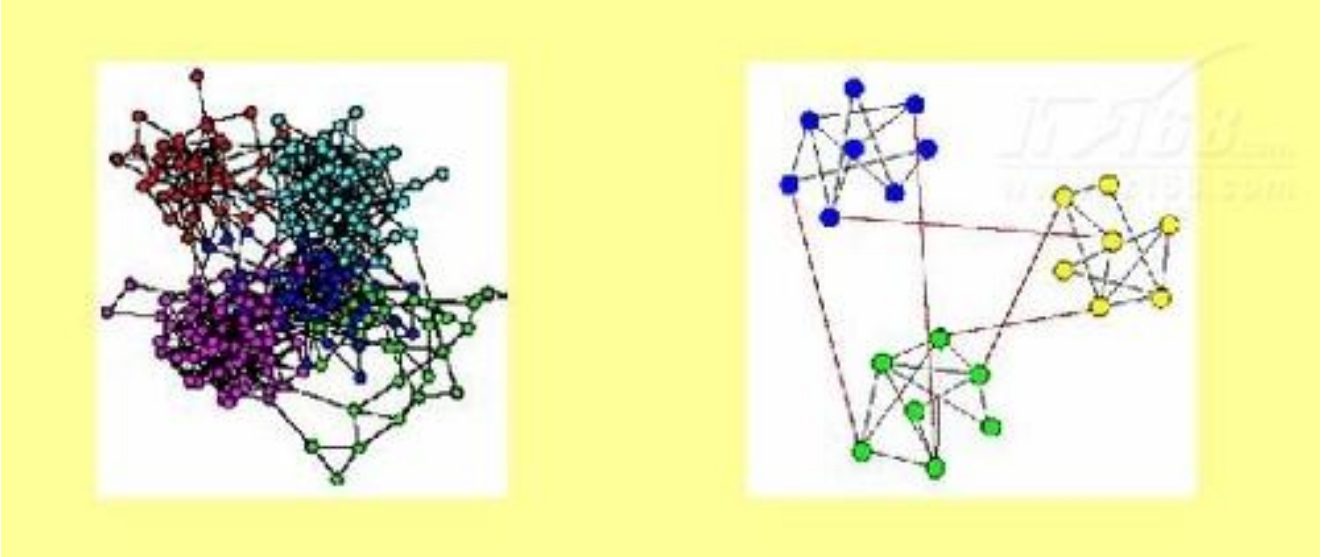




- 聚类中没有任何指导信息，完全按照数据的分布进行类别划分，是一种无监督的学习

聚类

为什么要聚类



- 客户分割（segmentation）是一种发现用户特性的方法。
- 将一个基于数据的客户信息分组；从而给你一个客户信息的概况，这可以直接转化为针对不同客户的营销策略。

聚类

应用领域

- **经济领域：**
  - 帮助市场分析人员从客户数据库中发现不同的客户群
  - 对住宅区进行聚类，确定自动提款机ATM的安放位置
  - 股票市场板块分析，找出最具活力的板块龙头股
  - 企业信用等级分类
- **生物学领域：**
  - 推导植物和动物的分类；
  - 对基因分类，获得对种群的认识
- **其他：**
  - 作为其他数学算法的预处理步骤，获得数据分布状况

聚类

原理

- 聚类分析中“类”的特征：
  - 聚类所说的类不是事先给定的，而是根据数据的相似性和距离来划分；
  - 聚类的数目和结构都没有事先假定

- 聚类方法的目的是寻找数据中：
  - 潜在的自然分组结构
  - 感兴趣的关系

K-means-算法概述

Q1：K是什么？
A1：K是聚类算法当中类的个数。

Q2：means是什么？
A2：means是均值算法。

- Summary：K-means是采用均值算法把数据分成K个类的硬聚类算法！
- 对于连续型属性具有较好的聚类效果，不适合处理离散型属性。

k-means-评价标准

- 一个好的聚类方法要能产生高质量的聚类结果—簇，这些簇要具备以下两个特点：
  - 高的簇内相似性
  - 低的簇间相似性

基本思想：属于划分方法（partitioning method），通过迭代把数据集划分为不同的类别（或称簇），使得评价聚类性能的准则函数达到最优，使得每个聚类类内紧凑，类间独立。

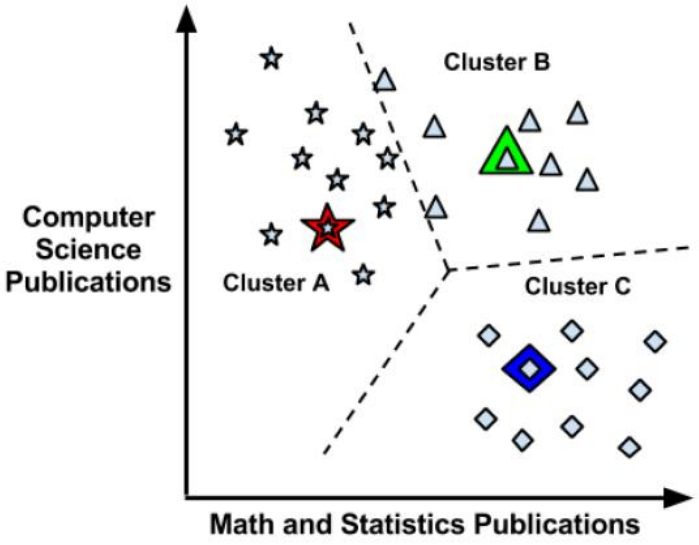
算法概述-基本流程

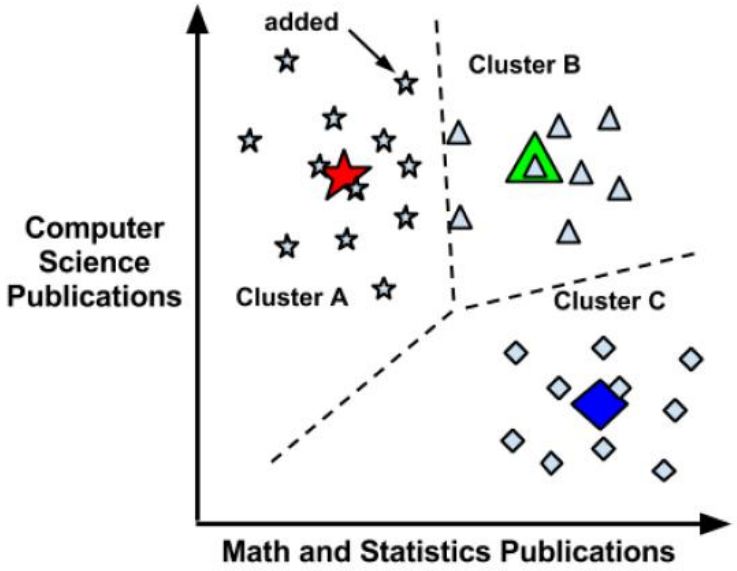
1. 随机抽取k个点作为初始聚类的中心，由各中心代表各聚类

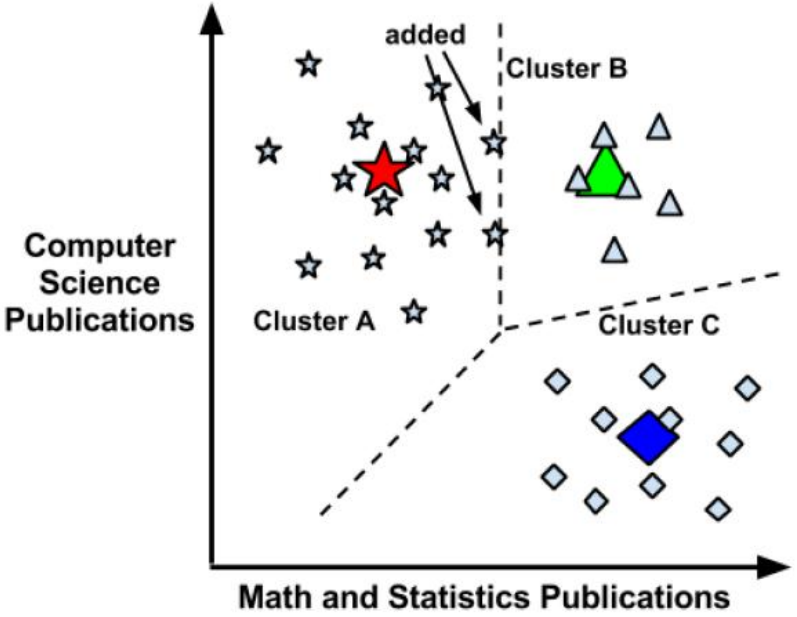
2.计算所有点到这k个中心的距离，并将点归到离其最近的聚类

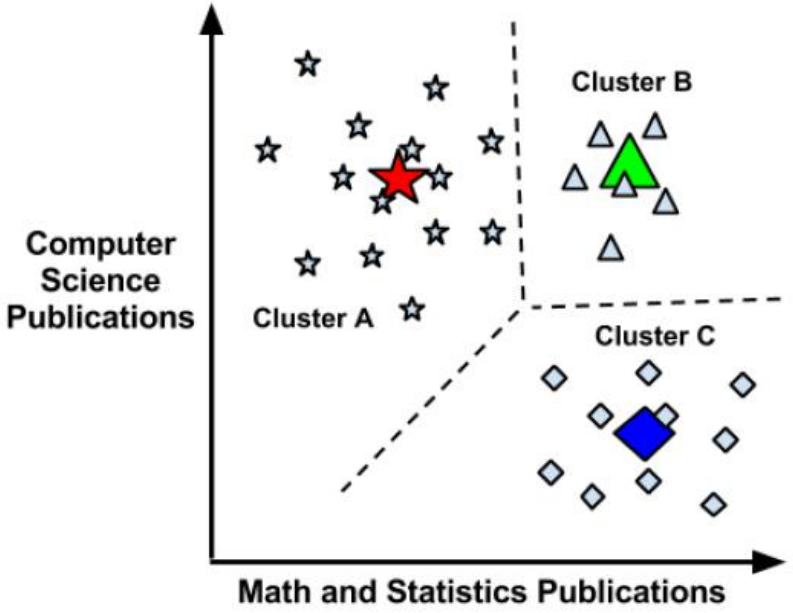
3. 调整聚类中心，即将聚类的中心移动到聚类的几何中心（即平均值）

4.重复第2、3步直到聚类的中心不再移动，此时算法收敛


Cluster B
Computer Science Publications
Cluster A
Cluster C
Math and Statistics Publications


added
Cluster B
Computer Science Publications
Cluster A
Cluster C
Math and Statistics Publications


added
Cluster B
Computer Science Publications
Cluster A
Cluster C
Math and Statistics Publications


Cluster B
Computer Science Publications
Cluster A
Cluster C
Math and Statistics Publications

K-means-案例

像元	波段	
	B1	B2
A	10	30
B	8	32
C	22	18
D	30	10
E	32	12

How about C ?

（1）确定类别数为2（ABC，DE），计算两个类的中心坐标

	类中心坐标	
类	B1	B2
ABC	13	27
DE	31	11

（2）计算每个像元到类中心的欧氏距离，并将每个像元重新分配给最近的一类。若类中像元

A到两个类的平均距离

D^2（A，ABC）=（10-13）^{2}+（30-27）^{2}=18

D^2（A，DE）=（10-31）^{2}+（30-11）^{2}=802

B到两个类的平均距离

D^2（B，ABC）=（8-13）^{2}+（32-27）^{2}=50

D^2（B，DE）=（8-31）^{2}+（32-11）^{2}=970

D到两个类的平均距离

D^2（D，ABC）=（30-13）^{2}+（10-27）^{2}=578

D^2（D，DE）=（30-31）^{2}+（10-11）^{2}=2

E到两个类的平均距离

D^2（E，ABC）=（32-13）^{2}+（12-27）^{2}=586

D^2（E，DE）=（32-31）^{2}+（12-11）^{2}=2

结论1：C应重新分配到DE所在类

重复步骤1、2；

	新的类中心坐标	
类	B1	B2
AB	9	31
CDE	28	13

A到两个类的平均距离

D^2（A，AB）=（10-9）^{2}+（30-31）^{2}=**2**

D^2（A，CDE）=（10-28）^{2}+（30-13）^{2}=613

D到两个类的平均距离

D^2（D，AB）=（30-9）^{2}+（10-31）^{2}=882

D^2（D，CDE）=（30-28）^{2}+（10-13）^{2}=**13**

B到两个类的平均距离

D^2（B，AB）=（8-9）^{2}+（32-31）^{2}=**2**

D^2（B，CDE）=（8-28）^{2}+（32-13）^{2}=761

E到两个类的平均距离

D^2（E，AB）=（32-9）^{2}+（12-31）^{2}=890

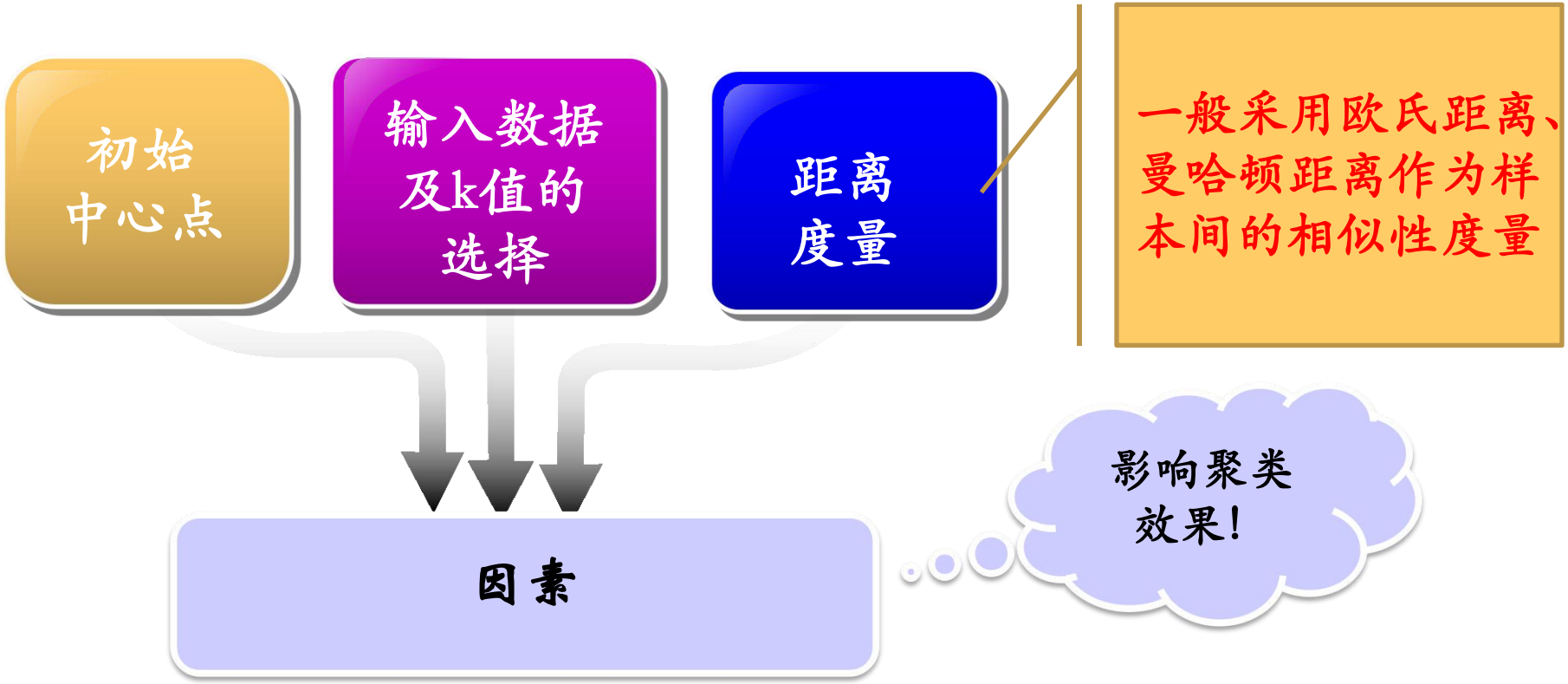
D^2（E，CDE）=（32-28）^{2}+（12-13）^{2}=**17**

结论：所有像元不再重新分类

各类平均距离汇总

	A	B	C	D	E
AB	2	2	338	882	890
CDE	613	761	61	13	17

K-means-主要因素


初始
中心点
输入数据
及k值的
选择
距离
度量
一般采用欧氏距离、
曼哈顿距离作为样
本间的相似性度量
因素
影响聚类
效果!

K-means-主要因素

初始中心点

1. 随机选点的方法
2.凭借经验选取有代表性的点
3.基于取样的方法确定
4.基于密度的选择方法

选择k的值

1. 凭检验直观选择k
2.按密度大小选代表点确定k
3.使距离度量方法值最小的k
4.最大最小距离法确定

K-means-优缺点

优缺点

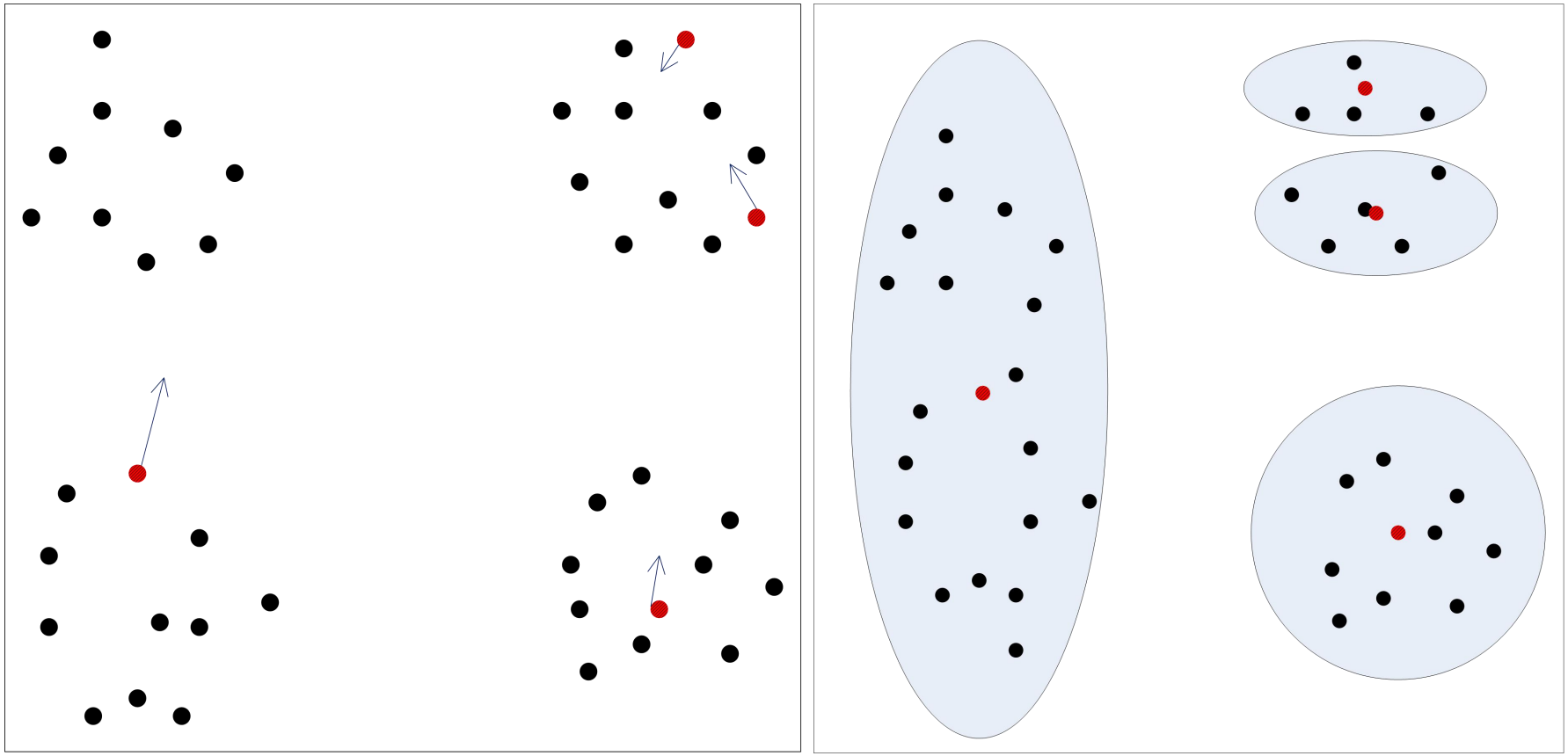
主要优点

1. 思想简单易行
2. 时间杂度接近线性
3. 对大数据集，具有高效性和可伸缩性

主要缺点

1. 依赖于初始均值的选择
2. 须事先给定聚类数k值
3. 对噪声和孤立数据敏感

K-means的优缺点



初始化4个类别中心

左侧的全体数据仅与第一个类别中心相似

用Kmeans算法探寻青少年市场细分

Step1：数据收集

- **数据来源：**圣母大学对于青少年的身份进行社会研究时编制的。完整的数据集可以冲Packt出版社网站获得。

- **研究目的：**与朋友在社交网站进行交互已经成为世界各地青少年的成年礼。营销者希望向他们销售小吃、饮料、电子产品和卫生用品。他们努力对有相似口味的青少年进行分类，从而避免将广告投给哪些正在销售的产品不感兴趣的青少年。

- **数据基本信息：**

- 数据采集了2006-2009四个高中毕业年份，将社交网络服务页面内容划分成单词，在出现最多的前500个单词中，36个单词被用来代表5大类兴趣，即课外活动（extracurricular activity）、时尚（fashion）、宗教（religion）、浪漫（romance）和反社会行为（antisocial behavior）；36个单词包括football, sexy, kissed, bible, shopping, death, drugs等等。
- 对每个人来说，最终的数据集表示每个单词出现在个人社交网络服务文件中的次数。

Step2：探索和准备数据

数据包括了30000名青少年的40个特征数据，其中前4个是基本信息，后面的36个数据是

```
> teens <- read.csv("snsdata.csv")

> str(teens)
'data.frame':   30000 obs. of  40 variables:
 $ gradyear    : int  2006 2006 2006 2006 2006 2006 2006 2006 ...
 $ gender      : Factor w/ 2 levels "F","M": 2 1 2 1 NA 1 1 2 ...
 $ age         : num  19 18.8 18.3 18.9 19 ...
 $ friends     : int  7 0 69 0 10 142 72 17 52 39 ...
 $ basketball  : int  0 0 0 0 0 0 0 0 0 0 ...
```

前4个是基本信息

Step2：探索和准备数据

- 通过检查数据，发现有很多缺失数据和不合理的数据

```
> table(teens$gender, useNA = "ifany")
    F     M  <NA>
22054  5222  2724

> summary(teens$age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  3.086  16.310  17.290  17.990  18.260 106.900    5086

> teens$age <- ifelse(teens$age >= 13 & teens$age < 20,
                      teens$age, NA)

> summary(teens$age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  13.03   16.30   17.26   17.25   18.22   20.00    5523
```

- 对缺失数据进行编码
- 性别的缺失可以编译为“unknown”；age的缺失可以用已有观测的均值填补。

Step3：基于数据训练模型

- 输入的关键参数是：包含需要的聚类的数据矩阵；指定的K。
- 我们仍然需要注意要对数据进行标准化。

聚类语法

应用 stats 添加包中的函数 kmeans()

建立模型：

```
myclusters <- kmeans( mydata, k)
```

- mydata：是包含需要聚类的实例的一个矩阵或者数据框
- k：给定需要的类的个数

该函数返回一个含有聚类结果的聚类对象。

检验聚类结果：

- myclusters$cluster 是 kmeans() 函数所给出的类成员向量
- myclusters$centers 是含有每个类组合和每一个特征的均值的一个矩阵
- myclusters$size 给出每一个类中实例的个数

例子：

```
teen_clusters <- kmeans(teens, 5)
teens$cluster_id <- teen_clusters$cluster
```

Step4：评估模型的性能

- 评估聚类的结果具有一定的主观性，视研究的预期目的来决定。
- 评估一个类是否有用的最基本方法之一就是检查落在每一组的案例数，案例数太多或者太少，这个类就没什么作用。

```
> teen_clusters$size
[1]  3376   601  1036  3279 21708
```

最小的类包含了2%的个体

最大的类包含了72%的个体

没有只包含一两个个体的类！

Step4：评估模型的性能

- 考察每一类的个体，在各个特征上的均值（即重心）

```
> teen_clusters$centers
   basketball     football       soccer     softball
1  0.02447191  0.10550409  0.04357739 -0.02411100
2 -0.09442631  0.06927662 -0.09956009 -0.04697009
3  0.37669577  0.38401287  0.14650286  0.15136541
4  1.12232737  1.03625113  0.53915320  0.87051183
5 -0.18869703 -0.19317864 -0.09245172 -0.13366478

   volleyball     swimming cheerleading     baseball
1  0.04803724  0.31298181   0.63868578 -0.03875155
2 -0.07806216  0.04578401  -0.10703701 -0.11182941
3  0.09157715  0.24413955   0.18678448  0.28545186
4  0.78664128  0.11992750   0.01325191  0.86858544
5 -0.12850235 -0.07970857  -0.10728007 -0.13570044
```

正负号，仅仅表示高于或低于平均水平

Cluster 1 (N = 3,376)	Cluster 2 (N = 601)	Cluster 3 (N = 1,036)	Cluster 4 (N = 3,279)	Cluster 5 (N = 21,708)
swimming cheerleading cute sexy hot dance dress hair mall hollister abercrombie shopping clothes	band marching music rock	sports sex sexy hot kissed dance music band die death drunk drugs	basketball football soccer softball volleyball baseball sports god church Jesus bible	???
Princesses	Brains	Criminals	Athletes	Basket Cases

Step5：模型性能的提高

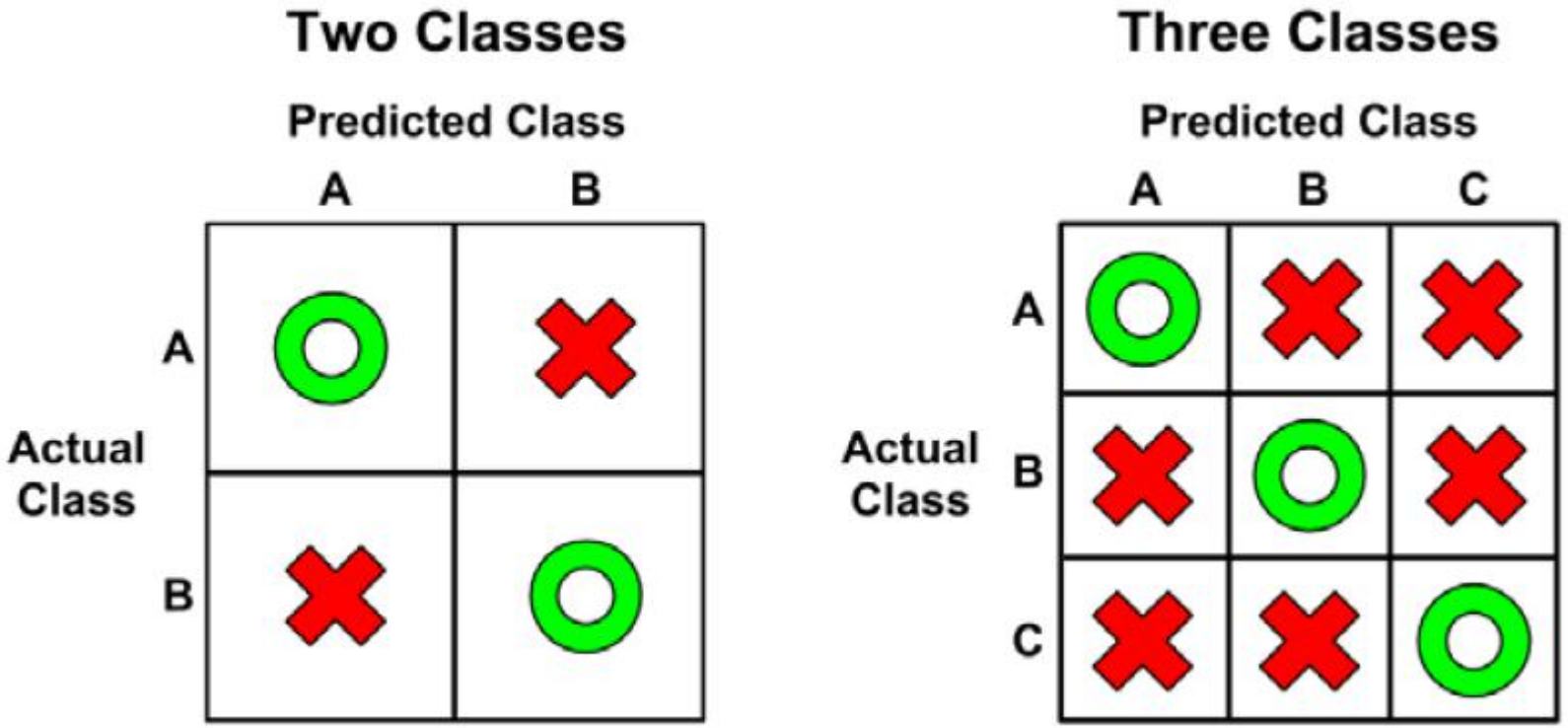
- 聚类创造了新的信息，可以把聚类创造的分类信息加载到原始数据上，可以做进一步的分析。
- 对于30000个青少年而言，每个人都贴上了分类标签。
- 可以进一步考察每个组的性别、朋友数量这些信息。

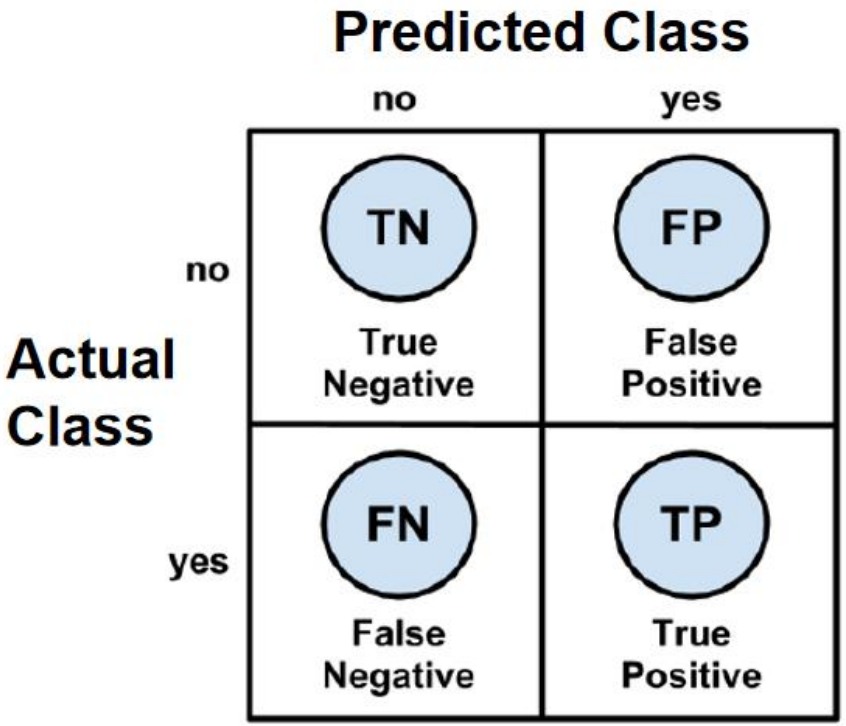
比如，考察每一类的朋友数量：公主类最高、其次是运动员和聪明组，再次是罪犯类，无特征组有27.9。

类似地，还可以考察其他信息，然后进行深度的挖掘。

Part III 模型性能的评价

混淆矩阵


Two Classes
Predicted Class
A
B
Actual Class
A
B
Three Classes
Predicted Class
A
B
C
Actual Class
A
B
C



准确率

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

错误率

$$\text{error rate} = \frac{FP + FN}{TP + TN + FP + FN} = 1 - \text{accuracy}$$

Kappa统计量

- Kappa统计量是一种衡量分类进度的指标，其取值在-1~1之间，一般落在0~1范围类。衡量了预测值和真实值之间的完全一致性。
- Kappa统计量一般基于混淆矩阵来计算，有很多不同的定义方式，常见的定义方式：

$$\text{Kappa} = \frac{P_0 - P_e}{1 - P_e},$$

$$P_0 = \frac{\text{每一类正确分类数}}{\text{总样本数}}$$

$$P_e = \frac{\text{每一类正确分类数} \times \text{预测正确分类数}}{\text{总样本数}^2}$$

预测类别 \ 实际类别	A	B	C
A	**239.0**	21.0	16.0
B	16.0	**73.0**	4.0
C	6.0	9.0	**280.0**

Kappa系数示例

上图就是个混淆矩阵，

其中，$p_o = \frac{239 + 73 + 280}{664} = 0.8916.$

$$p_e = \frac{261 \times 276 + 103 \times 93 + 300 \times 295}{664 \times 664} = 0.3883.$$

因此：

$$k = \frac{0.8916 - 0.3883}{1 - 0.3883}.$$

- 很差的一致性：小于 0.2。
- 尚可的一致性：0.2 ~ 0.4。
- 中等的一致性：0.4 ~ 0.6。
- 不错的一致性：0.6 ~ 0.8。
- 很好的一致性：0.8 ~ 1。

如上例中：**Kappa=0.823**，说明预测和真实值的一致性较好

灵敏度与特异性

- **灵敏度（sensitivity）**：阳性样本被正确分类的比例。

$$\text{sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **特异性（specificity）**：阴性样本被正确分类的比例。

$$\text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

- 灵敏度和特异性取值范围在0~1，越接近1的值越好。灵敏度和特异性需要进行权衡。

精确度和回溯精确度

- **精确度（precision）**：定义为阳性的占所有预测为阳性的比例。

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **回溯精确度（recall）**：真阳性与阳性总数的比例（和特异性的公式是一样的。

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- 与灵敏度和特异度之间的权衡相似，大多数的真实问题，很难建立一个同时具有很高精确度和回溯精确度的模型，非常具有挑战性。

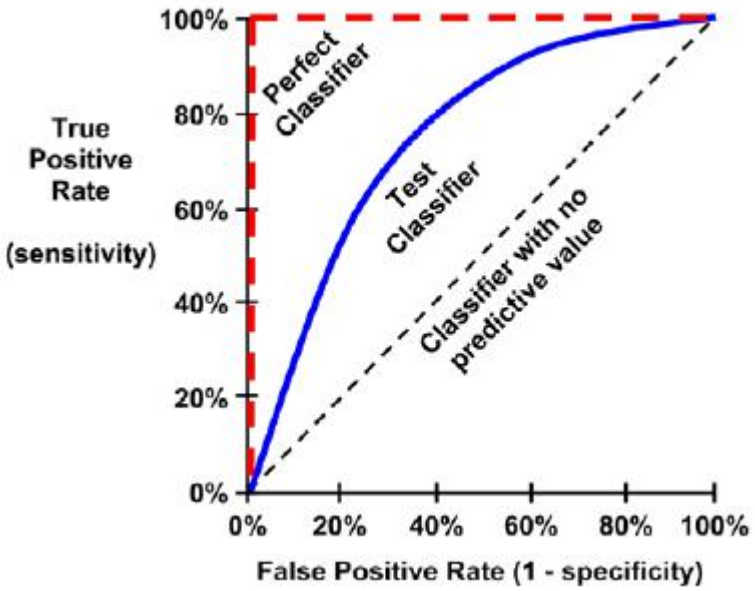
F度量

- **F度量（F-measure）：**定义将精确度和回溯精确度合并成一个但一值的度量值，也称F1计分或者F计分

$$\text{F-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{recall} + \text{precision}} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}}$$

- F度量将模型的性能指标变成一个单一的值，提供了一种便利的方式来比较多个模型的好坏。这里对精确度和回溯精确度设置了同样的权重，也可以用不同的权重。

ROC (Receiver Operating Characteristic)曲线



- **ROC曲线：**纵轴表示真阳性的比例（灵敏度），横轴表示假阳性的比例（1-特异度），也称灵敏度/特异性图。
- ROC曲线上的点，表示不同假阳性阈值上，真阳性的比例。

- ROC曲线下面积（Area Under the ROC，AUC）：AUC将ROC图看成二维正方形，测量ROC曲线下面的面积。

- 0.9 ~ 1.0 = A（优秀）。
- 0.8 ~ 0.9 = B（良好）。
- 0.7 ~ 0.8 = C（一般）。
- 0.6 ~ 0.7 = D（很差）。
- 0.5 ~ 0.6 = F（无法区分）。