北京郵電大學
BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

Data Analytics with R
语言数据分析

方以类聚、物以群分

艾新波 / 2018 • 北京

课程体系

- R语言数据分析
  - 上部：论道 
    - 第1章 气象万千、数以等观
    - 第2章 所谓学习、归类而已
    - 第3章 格言联璧话学习
    - 第4章 源于数学、归于工程
  - 中部：执具 
    - 第5章 工欲善其事必先利其器
    - 第6章 基础编程
    - 第7章 数据对象
  -
    - 第8章 人人都爱tidyverse
    - 第9章 最美不过数据框
  - 下部 博术 
    - 第10章 观数以形
    - 第11章 相随相伴、谓之关联
    - 第12章 既是世间法、自当有分别
    - 第13章 方以类聚、物以群分
    - 第14章 庐山烟雨浙江潮

基于划分的聚类方法

基于划分的方法是聚类分析中最简单、最基本的版本：

试图发现用户指定个数（k）的簇（由质心代表）

形式化表述如下：

给定m 个对象的数据集D以及要生成的簇数k，划分算法把数据对象分成k个簇，$C_1, C_2, \cdots, C_k$，使得$1 \leq i, j \leq k$，$C_i \subset D$，$C_i \cap C_j = \Phi$

目标函数用来评估划分的质量，使得簇内对象相互相似，而与其它簇中对象相异：簇内高相似性、簇间低相似性

基于划分的聚类方法

目标函数——误差平方和Sum of Squared Error, SSE

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} dist\left(c_i, x\right)^2$$

这个目标函数试图**使得生成的结果簇尽量紧凑和独立**

要通过蛮力的方法，枚举或者说遍历所有可能的划分，并计算相应的SSE值，即便是$k = 2$，也是一个NP问题

解决问题一个常用的方法是$k - Means$方法

k-Means算法

输入: k——簇的数目； D——包含m个对象的数据集

输出: k个簇的集合

1: 从D中随机选择k个对象作为初始簇中心

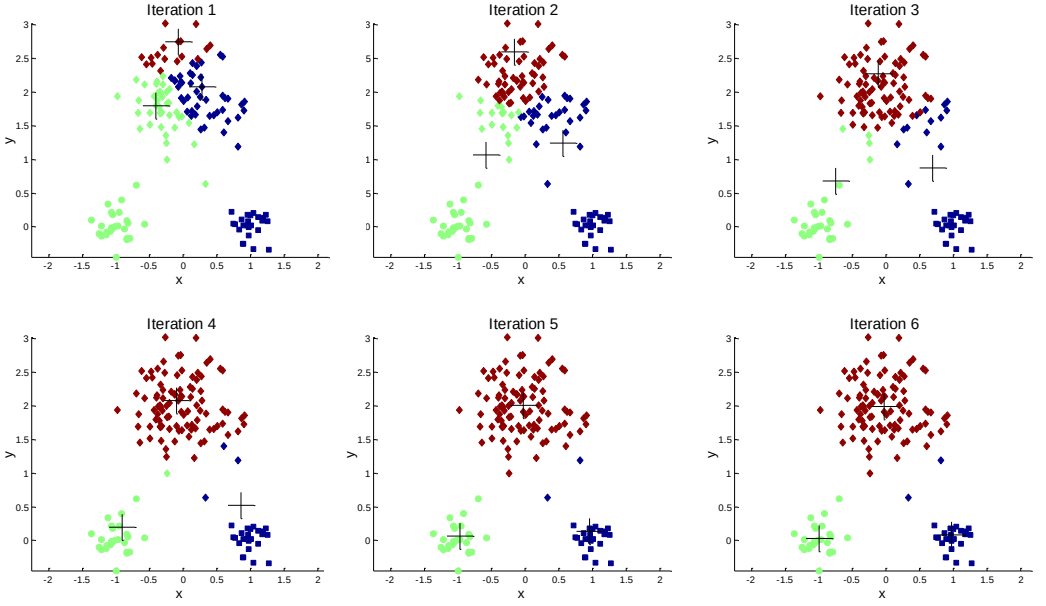
2: Repeat

3: 根据离簇中心的远近，将每个对象分配到相应的簇

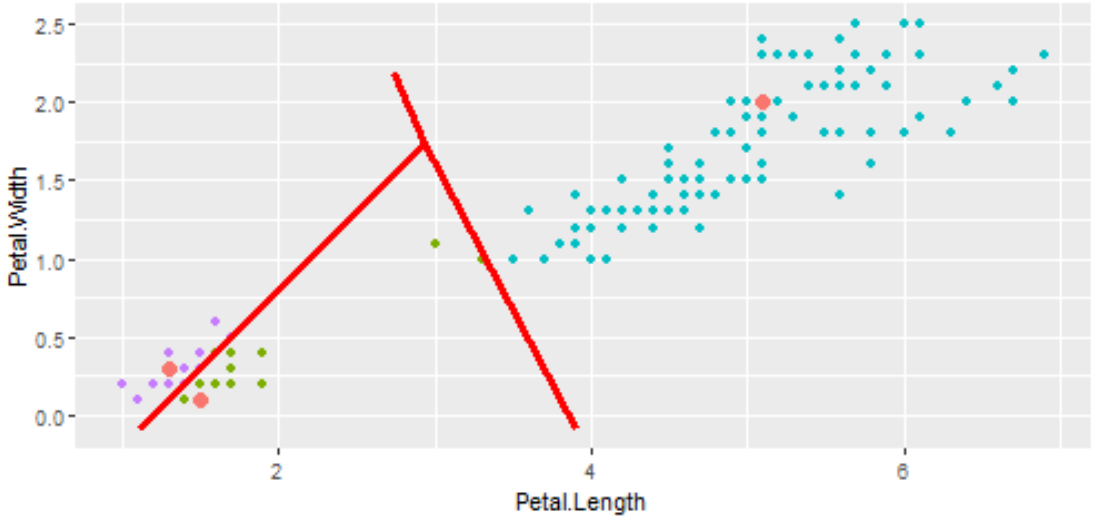
4: 更新簇均值，即重新计算每个簇中对象的均值作为簇中心

5: Until 不再发生变化

算法示意图


Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
x
y

算法示意图



算法实现

Partitioning based Clustering: partition the data into k groups first and then try to improve the quality of clustering by moving objects from one group to another

`stats::kmeans()` perform k-means clustering on a data matrix

`fpc::kmeansruns()` call kmeans for the k-means clustering method and includes estimation of the number of clusters and finding an optimal solution from several starting points

`fpc::kmeansCBI()` interface function for kmeans

算法实现：基本参数设置

```
scores <- cjb %>%
  select(yw:sw)
set.seed(2012)
imodel <- kmeans(scores, centers = 2)
names(imodel)
#> [1] "cluster"      "centers"      "totss"
#> [4] "withinss"     "tot.withinss" "betweenss"
#> [7] "size"         "iter"         "ifault"
```

算法实现：模型解释

```
imodel$cluster
#> [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1
#> [24] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
#> [47] 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1
.....
#> [714] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
#> [737] 2 2 2 2 2 2 2 2 2 1 2 2 2 1 1 1 1 1 1 1 1 1 1
#> [760] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

算法实现：模型解释

```
imodel$centers
#>      yw       sx       wy       zz       ls
#> 1 85.07616 77.06623 83.24172 90.54967 84.85430
#> 2 88.85169 92.03178 90.24153 93.46822 91.88559
#>      dl       wl       hx       sw
#> 1 90.08940 71.22185 85.64901 79.58609
#> 2 94.91949 87.58475 95.54661 90.72034
```

算法实现：模型解释

```
imodel$totss
#> [1] 431975.9
imodel$withinss
#> [1] 168392.4 105535.6
imodel$tot.withinss
#> [1] 273928
imodel$betweenss
#> [1] 158047.9
```

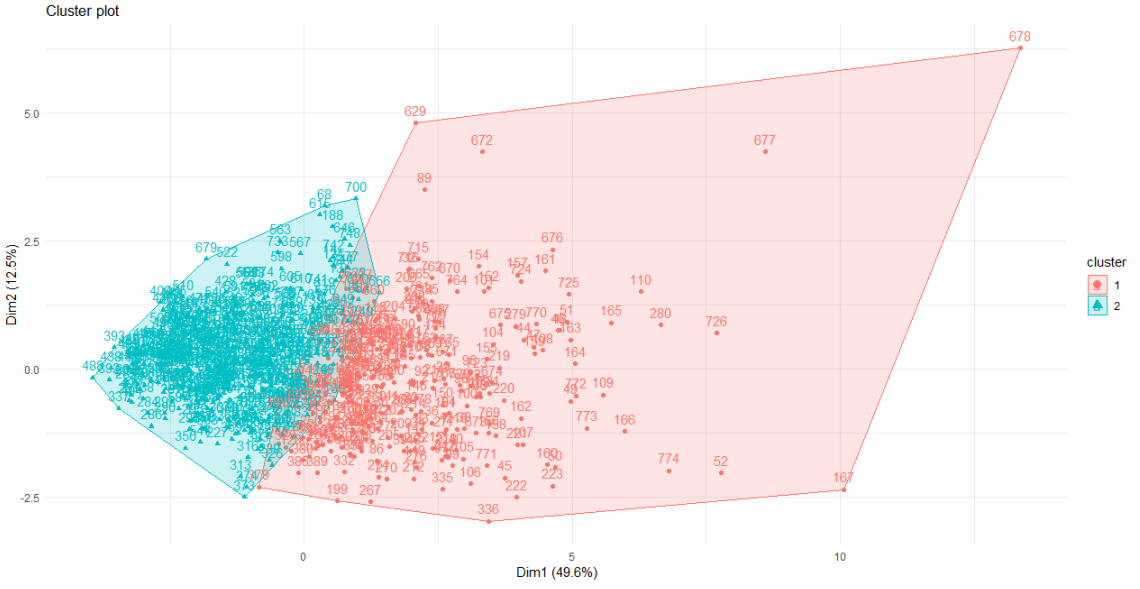
算法实现：模型解释

```
#手工实现一次代码
global_center <- apply(scores, 2, mean)
total_SS <- sum(apply(scores, 1, function(x) {
  sum((x - global_center)^2)
}))
total_SS
#> [1] 431975.9
```

算法实现：绘制聚类效果图

```
#绘制聚类效果图
library(factoextra)
fviz_cluster(imodel,
             data = scores,
             ellipse.type = "convex") +
  theme_minimal()
```

算法实现：绘制聚类效果图



算法实现：模型评估与k值选择

```
library(fpc)
kmeans_results <- kmeansruns(scores, criterion="asw")
kmeans_results
#> Available components:
#>   [1] "cluster"      "centers"      "totss"
#> [4] "withinss"     "tot.withinss" "betweenss"
#> [7] "size"         "iter"         "ifault"
#> [10] "crit"         "bestk"
```

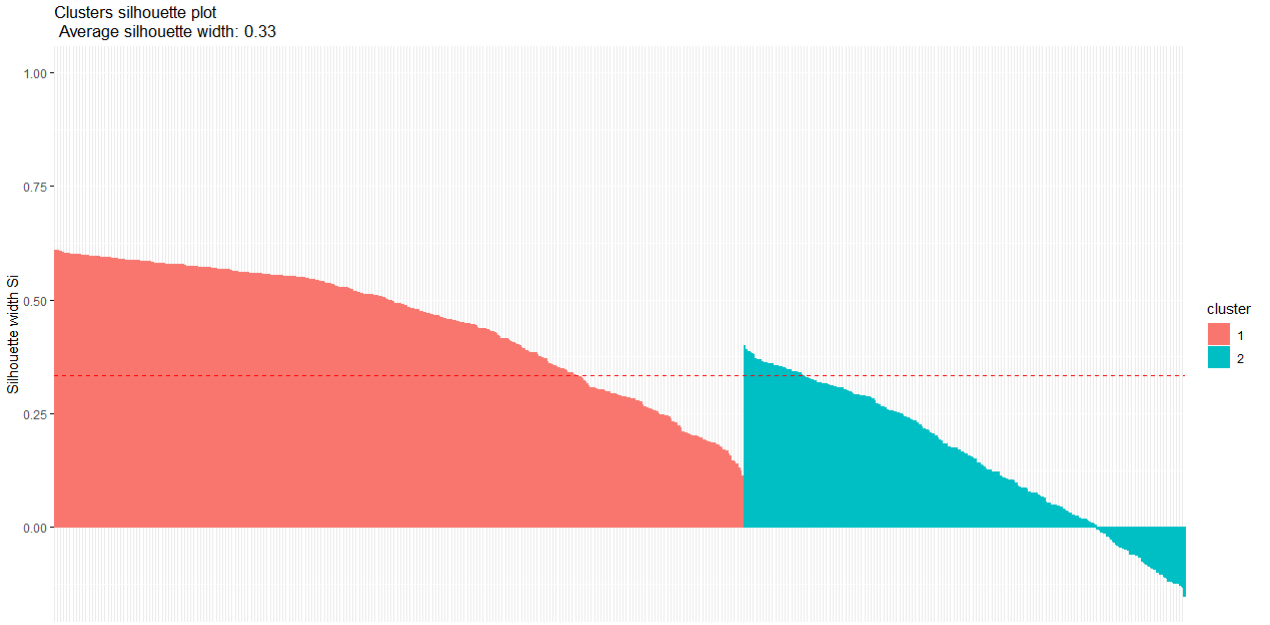
算法实现：模型评估与k值选择

```
kmeans_results$bestk
#> [1] 2
kmeans_results$crit
#> [1] 0.0000000 0.3330389 0.2490188 0.2453507
#> [5] 0.1889768 0.1652665 0.1603259 0.1579961
#> [9] 0.1552047 0.1401957
```

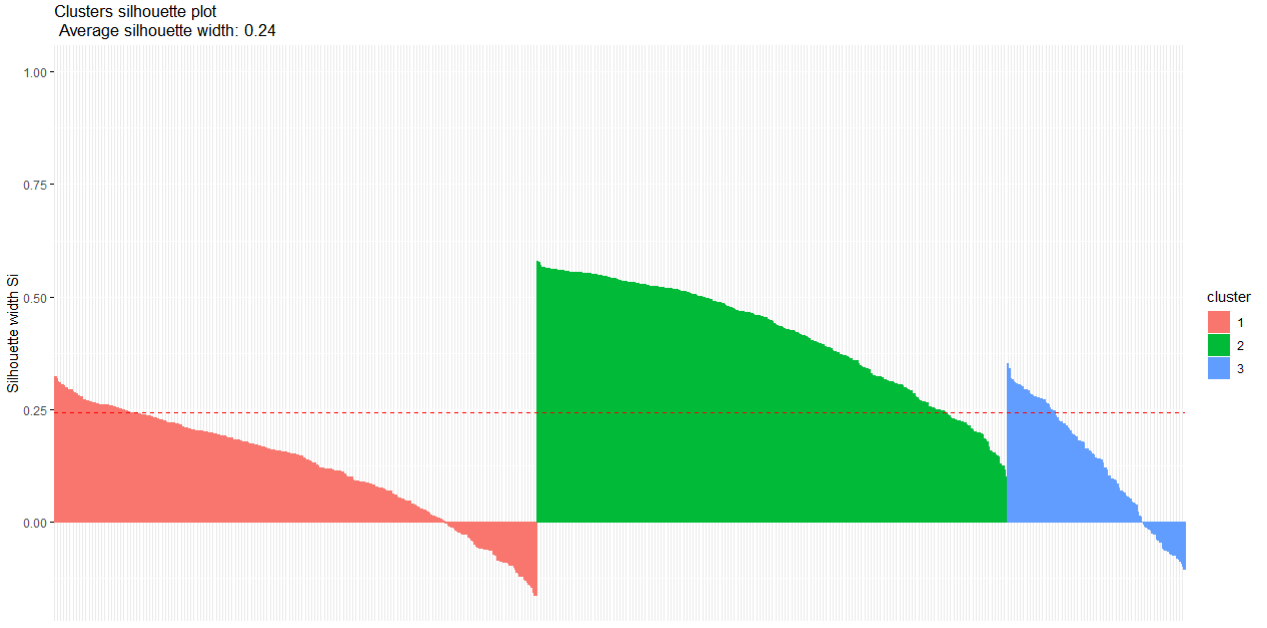
算法实现：模型评估与k值选择

```
require(cluster)
scores_dist <- dist(scores)
imodel2 <- kmeans(scores, 2)
cluster_idx2 <- imodel2$cluster
#计算轮廓系数
k2_silhouette <- silhouette(cluster_idx2, scores_dist)
#绘制轮廓系数
fviz_silhouette(k2_silhouette)
```

算法实现：模型评估与k值选择



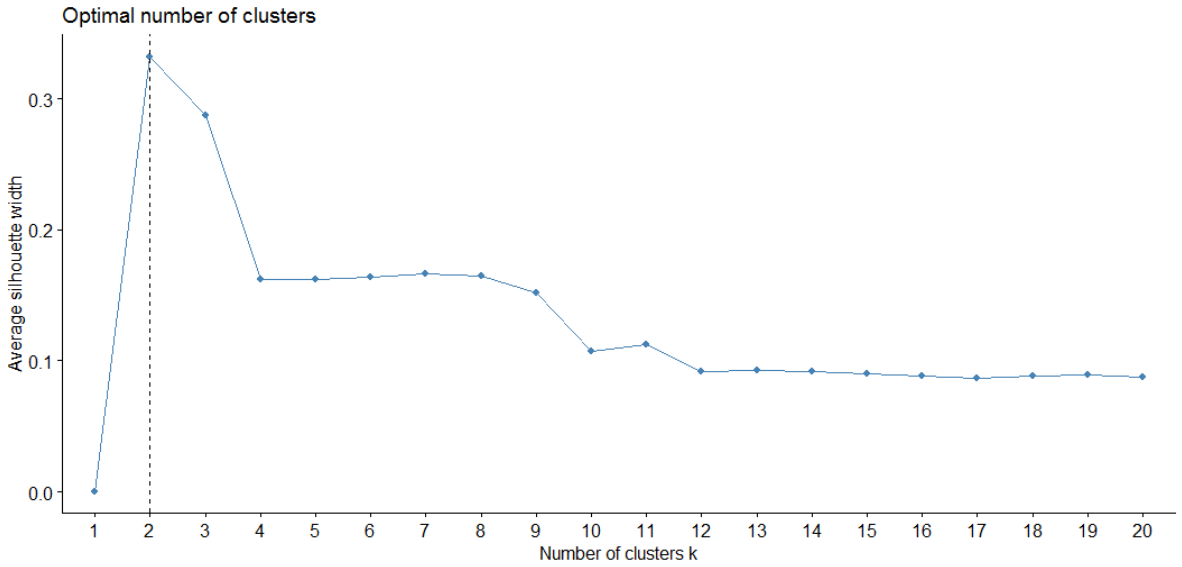
算法实现：模型评估与k值选择



算法实现：模型评估与k值选择

```
library(factoextra)
fviz_nbclust(scores,
             kmeans,
             method = "silhouette",
             k.max = 20) +
  geom_vline(xintercept = 2, linetype = 2)
```

算法实现：模型评估与k值选择



算法实现：与实际类标签的比较

```
#与实际类标签相比较
min(Metrics::ce(cjb$wlfk,
                c( "理科", "文科")[imodel$cluster]),
    1- Metrics::ce(cjb$wlfk,
                   c( "理科", "文科")[imodel$cluster]))
#> [1] 0.3540052
```

谢谢聆听

Thank you

教师个人联系方式

艾新波

手机：13641159546

QQ： 23127789

微信：13641159546

E-mail: 13641159546@126.com

axb@bupt.edu.cn

地址：北京邮电大学科研楼917室

课程网址：https://github.com/byaxb/RDataAnalytics