文本聚类

教师：邱锡鹏 微博：@邱锡鹏

助教：施展、龚经经

复旦大学 计算机学院 大数据学院

文本聚类方法可以分为静态聚类和动态聚类，静态聚类方法包括Top-down方法和Bottom-up方法，动态聚类（Online clustering）需要判断每个新加入的样本属于已有的类还是一个新类。

2.1 K-mean聚类（Top-down）

K-means聚类前提：样本之间相似度能够计算

1. 随机在图中取K个种子点。
2. 然后对图中的所有点求到这K个种子点的距离，假如点P_i离种子点S_i最近，那么P_i属于S_i点群。
3. 接下来，我们要移动种子点到属于他的“点群”的中心。
4. 然后重复第2）和第3）步，直到种子点没有移动。

K-means缺点：

1. 需提前确定聚类数目
2. 对初始选取的点很敏感
3. 属于硬聚类（每个样本只能属于一类）

2.2 布朗聚类（Bottom-Up）

2.2.1 预备知识

熵

若X是一个离散型随机变量，取值空间为R，其概率分布为$p(x)=P(X=x),x\in R$。那么，X的熵$H(X)$定义为

$$H(X)=-\sum_{x\in R}p(x)log_2p(x)$$

其中约定$0log_20=0$，通常将$log_2p(x)$写作$logp(x)$

熵又称为自信息，可以视为描述一个随机变量的不确定性的数量。

联合熵和条件熵

若X,Y是一对离散型随机变量$X,Y\sim p(x,y)$，则X,Y的联合熵$H(X,Y)$

$$H(X,Y)=-\sum_{x\in X}\sum_{y\in Y}p(x,y)logp(x,y)$$

联合熵实际就是描述一对随机变量平均所需的信息量

给定随机变量X的情况下，随机变量Y的条件熵如下：

$$H(Y|X)=-\sum_{x\in X}\sum_{y\in Y}p(x,y)logp(y|x)$$

条件熵和联合熵的关系：

$$H(X,Y)= H(Y|X)+H(X)$$

推广到一般情况：

$$H(X_1,X_2,\ldots,X_n)=H(X_1)+H(X_2|X_1)+\ldots+H(X_n|X_1,\ldots,X_{n-1})$$

互信息

$$H(X,Y)=H(X)+H(Y|X)=H(Y)+H(X|Y)$$

可得

$$H(X)-H(X|Y)=H(Y)-H(Y|X)$$

这个差称为X和Y 的**互信息**，记作$I(X;Y)$，可得$I(X;Y)=H(X)-H(X|Y)$

$I(X;Y)$反映的是在知道了Y的值以后X的不确定性的减少量。

$$I(X;Y)=\sum_{x,y}p(x,y)log\frac{p(x,y)}{p(x)p(y)}$$

2.2.2 布朗聚类：

布朗聚类是一种自底而上的层次聚类算法，基于n-gram模型和马尔科夫链模型，是一种硬聚类，每个词都在且只在唯一的一个类中。

w是词，c是类，不同于词性标注，此处c是未知的。

布朗聚类的输入是一个语料库，这个语料库是一个词序列，输出是一个二叉树，树的叶子节点是一个个词，树的中间节点是我们想要的类（中间结点作为根节点的子树上的所有叶子为类中的词）。

$$\begin{aligned}Quality(C)&=\frac{1}{n}logP(w_1,\cdots,w_n)=\frac{1}{n}logP(w_1,\cdots,w_n,C(w_1),\cdots,C(w_n))\\&=\frac{1}{n}log\prod_{i=1}^{n}P(C(w_i)|C(w_{i-1}))P(w_i|C(w_i))\end{aligned}$$

$C(w_0)$是种特殊的'Start'类，我们要找到这样的分类方式C，使得$Quality(C)$尽可能大

$$\begin{aligned}Quality(C)&=\frac{1}{n}\sum_{i=1}^{n}P(C(w_i)|C(w_{i-1}))P(w_i|C(w_i))\\&=\sum_{w',w}\frac{n(w,w')}{n}logP(C(w')|C(w)))P(w'|C(w'))\\&=\sum_{c,c'}\frac{n(c,c')}{n}log\frac{n(c,c')}{n(c)n(c')}+\sum_{w'}\frac{n(w')}{n}log\frac{n(w')}{n}\end{aligned}$$

$n(w)$是w在文本中出现次数，$n(w,w^{'})$是二元对$w,w^{'}$在文本中的出现次数，$n(c)=\sum_{w\in c}n(w)$，$n(c,c^{'})=\sum_{w\in c}n(w,w^{'})$

$$Quality(C)=I(C)-H$$

简单算法:

以k长度文本为例：每个字均为一类，词典大小为$|V|$，找到俩类，合并后使得互信息变得最大，重复合并步骤使得最后都归为一类，整个算法的时间复杂度为$O(|V|^5)$

优化算法：

开始设置一个参数m，比如$m=1000$，我们按照词汇出现的频率对其进行排序

然后把频率最高的m个词各自分到一个类中，对于第$m+1$到$|V|$个词进行循环：

1.对当前词新建一个类，我们现在又$m+1$个类了

2.从这$m+1$个类中贪心选择最好的两个类合并，现在我们剩下m个类

最后我们再做$m-1$词合并。算法时间复杂度$O(|V|*m^2)$