Lecture 5: K-means 聚类

2024.3.12

Lecturer: 丁虎 *Scribe:* 王浩宇

聚类法是数据处理与分析最基础的一类工具。聚类有很多方法，基于密度，基于类中心等等。最小生成树 Krusk 算法的生成过程也可以看成一个自底向上的层次聚类。每次加边的过程相当于将边的两个顶点合并成一类。

1 定义

Definition 1.1 (K-means)**.** 输入欧式空间中 n 个点的集合 $X = \{x_1, x_2, ..., x_n\} \in R^d$, 希望找到 k 个点 $C = \{c_1, ..., c_k\} \in R^d$, 使得 $\phi_X(C) = \sum_{x\in X} min_{c\in C} \|c - x\|_2^2$ 最小。

下面给出一些记号方便后面分析。

Definition 1.2. 对于任意子集 $A \subset X, \phi_A(C) = \sum_{x\in A} min_{c\in C} \|c - x\|_2^2$。

Definition 1.3. 记 $A_1, ..., A_k$ 为最优解 C 导出的类。其中 $A_i = \{x \in X|c_i = argmin_{c\in C} \|c - x\|_2^2\}$。

对于 K-means 问题即使 $k = 2, d = 2$ 都是 NP-hard 的。我们期望可以找到近似解，下面两种情况是相对较为简单的

- d 为常数时，Local Search 算法可以给出一个 PTAS。
- k 为常数时，Peeling 算法可以给出一个 PTAS

Peeling 算法先通过随机采样的方式估计出最大的类类中心的位置，以该类中心为圆心特定半径画一个球，删去其中的点。在剩下的点中继续找。

Definition 1.4 (重心)**.** 给定欧氏空间 R^d 中任意一个点集 S 其重心 $\mu(S) = \frac{1}{|S|}\sum_{x\in S} x$。其中 $|S|$ 为点集 S 中点的个数。

下面的技术性引理会很有帮助，后续会经常用到。

Lemma 1.5. *给定欧氏空间 R^d 中任意一个点集 S 以及 $p \in R^d$, 我们有*

$$\sum_{x\in S} \|x - p\|_2^2 = \sum_{x\in S} \|x - \mu(S)\|_2^2 + |S| \|\mu(S) - p\|_2^2$$

2 算法

Algorithm 1 Lloyd's Algorithm

均匀随机选取 k 个点作为 C 的初始化

while 算法未到稳定 **do**

  将 X 根据 C 中的 k 个类中心进行划分，得到 k 个类 $H_1, ..., H_k$.

  对每一个类 H_j, 更新类中心 $c_j \leftarrow \mu(H_j)$.

end while

该算法实际上并没有对近似比的保证，如果初始化的点选取很不好，算法可能会非常差，例如考虑一个长宽比非常大的矩形的四个顶点。下面是一个简单的改进。

Algorithm 2 K-means++

初始化 $C \leftarrow \{c_1\}, c_1$ 为 X 中随机选取的点。

for $j = 2, 3, ..., k$ **do**

  $p(x) \leftarrow \frac{min_{1\leq \ell \leq j-1} \|x-c_\ell\|_2^2}{\phi_X(C)}$.

  以概率 $p(x)$ 选取 X 中点 c_j 放入 C 中

end for

以上面得到的 C 作为 Loyed's Algorithm 算法中心的初始化，运行 Loyed's Algorithm.

该算法的想法是希望取到离所有中心最远的点的概率最大。为什么不直接使用贪心的思想？其实我们希望选出离最优解中心接近的点，这样的点往往并不是距离现有中心最远的点。直觉上最优解中心周围的点会比较稠密，这样会以更高概率取得离最优解中心接近的点。该算法的近似比期望为 $8\log(k)$.

下面介绍的算法是上面算法的一个变种, 很多时候我们并不确定需要将数据聚成多少类, 如果我们允许返回多于 k 个类，那么可以将 cost 显著降低，对于原本 k 聚类的算法能讲近似比改进到常数。

Algorithm 3 β Giteria approximation for K-means

初始化 $C \leftarrow \{c_1\}$, c_1 为 X 中随机选取的点。
for $j = 2, 3, ..., k, ..., \frac{16(k+\sqrt{k})}{3}$ **do**
  $p(x) \leftarrow \frac{min_{1\leq \ell \leq j-1}\|x-c_\ell\|_2^2}{\phi_X(C)}$.
  以概率 $p(x)$ 选取 X 中点 c_j 放入 C 中
end for
返回 C

该算法将返回大于 k 个类, 但是会将近似比变成常数。如果我们记对于 k 个类的最优解为 C_{opt} ,那么 $\phi_X(C) \leq 20\phi_X(C_{opt})$. 以至少常数概率成立。下面我们对 β Giteria approximation 进行分析。

为了方便，我们定义算法运行到第 i 步时类中心集合为 S_i. 初始化的集合为 $S_0 = \emptyset$。在第 i 步我们将最优解导出的类 $\{A_1, ..., A_k\}$ 分为两种集合，

$$Good_i = \{A_j | \phi_{A_j}(S_{i-1}) \leq 10\phi_{A_j}(C_{opt})\}.$$

$$Bad_i = \{A_1, ..., A_k\} \backslash Good_i.$$

Remark 2.1. 如果存在某一个时刻 j, $Bad_j = \emptyset$ 说明我们已经得到 10 近似比的解，

Remark 2.2. 从直觉上讲，我们希望对于 $i < j$ 有 $Good_i \subset Good_j$，这个算法才是有效的。

Lemma 2.3. 假设在第 i 步有两种事件

- $A = \{\phi_X(S_{i-1}) \leq 20\phi_X(C_{opt}).\}$
- $B = \{c_i \in Bad_i\}$

那么 $P[B|A^c] \geq \frac{1}{2}$.

Proof. 利用算法中的相关定义，计算选到 Bad_i 中点的概率。证明留作课后练习。 $\square$

Lemma 2.4. $\forall A_j \in Bad_i$, 定义其平均半径 $r = \sqrt{\frac{1}{|A_j|}\phi_{A_j}(C_{opt})}$. 同时定义 $d = \min_{y \in S_{i-1}} \|y - \mu(A_j)\|$。那么我们有 $d \geq 3r$.

Proof. 对于任意一个 $A_j \in Bad_i$, 根据定义我们有 $d = \min_{y \in S_{i-1}} \|y - \mu(A_j)\|$，并且

$$\begin{aligned} 10 \cdot \phi_{A_j}(C_{opt}) &< \phi_{A_j}(S_{i-1}) \\ &= \sum_{x \in A_j} \min_{y \in S_{i-1}} \|x - y\|_2^2 \\ &\leq \sum_{x \in A_j} \|x - y_0\|_2^2 \\ &= \Phi_{A_j}(C_{opt}) + |A_j| \cdot d^2 \end{aligned}$$

整理上式即可得到 $d \geq 3r$。 □

Lemma 2.5. 定义核心点集 $B_{A_j}(\alpha) = \{x \in A_j | \|x - \mu(A_j)\| \leq \alpha \cdot r\}$，其中 $0 \leq \alpha \leq 3$。$\forall b \in B_{A_j}(\alpha)$, $\Phi_{A_j}(S_{i-1} \cup \{b\}) \leq 10 \cdot \Phi_{A_j}(C_{opt})$.

Proof. 可以用 b 来替代 $\mu(A_j)$，可以得到

$$\Phi_{A_j}(S_{i-1} \cup \{b\}) \leq (1 + \alpha^2) \cdot \Phi_{A_j}(C_{opt}) \leq 10 \cdot \Phi_{A_j}(C_{opt})$$

□

Lemma 2.6. $|B_{A_j(\alpha)}| \geq (1 - \frac{1}{\alpha^2})|A_j|$

Proof.

$$\begin{aligned} \Phi_{A_j}(C_{opt}) &\geq \sum_{x \in A_j \setminus B(\alpha)} \|x - \mu(A_j)\|^2 \\ &\geq (|A_j - |B_{A_j}|) \cdot (\alpha r)^2 \\ &= (1 - \frac{|B_{A_j}|}{|A_j|}) \cdot \alpha^2 \cdot \Phi_{A_j}(C_{opt}). \end{aligned}$$

整理上式即可。 □

Lemma 2.7. 假设 x 为通过 *kmeans++* 采到的点，则 $\Pr[x \in B_{A_j}(\alpha) | A_j \in Bad_i, x \in A_j] = \frac{\Phi_{B_{A_j}}(S_{i-1})}{\Phi_{A_j}(S_{i-1})} \geq \frac{3-\alpha}{10} \cdot (1 - \frac{1}{\alpha^2})$

Proof. 由三角不等式,

$$\Phi_{B_{A_j}}(S_{i-1}) \geq |B_{A_j}| \cdot (d - \alpha r)^2$$

除此之外，

$$
\begin{aligned}
\Phi_{A_j}(S_{i-1}) &\leq \sum_{x\in A_j}\|x-y_0\|^2 \\
&\leq \sum_{x\in A_j}\|x-\mu(A_j)\|^2 + |A_j|\cdot\|\mu(A_j)-y_0\|^2 \\
&\leq \Phi_{A_j}(C_{opt}) + |A_j|\cdot \\
&\leq |A_j|(r^2+d^2).
\end{aligned}
$$

因此

$$
\begin{aligned}
\Pr[x\in B_{A_j}(\alpha)|A_j\in Bad_i, x\in A_j] &\geq \frac{|B_{A_j}|\cdot(d-\alpha r)^2}{|A_j|\cdot(r^2+d^2)} \\
&\geq \frac{(d-\alpha r)^2}{r^2+d^2}\cdot(1-\frac{1}{\alpha^2}) \\
&\geq \frac{1}{\alpha^2})\cdot\frac{(3-\alpha)^2}{10}
\end{aligned}
$$

□

通过上述引理，$S_i = S_{i-1}\cup\{x\}$，令 $\alpha\approx 1.44225$，则 $\Pr[\Phi_{A_j}\leq 10\Phi_{A_j}(C_{opt})|x\in A_j, A_j\in Bad_i]\geq 0.126$. (通过数值计算得到)。

回到 β Giteria approximation 的分析，通过引理 2.3 和上述不等式，$\Pr[|Bad_{i-1}|<|Bad_i| \mid |A^c]\geq 0.063$. 定义一个随机变量序列 $q_i, i=1,2,\cdots$

$$q_i = 1, \text{if } \ |Bad_{i+1}| = |Bad_i|$$

$$q_i = 0, \text{if } \ |Bad_{i+1}| = |Bad_i| - 1$$

因此，$\Pr[q_i=0|q_1,\cdots,q_{i-1}] = 0.063 \triangleq p$ 并且 $E[q_i|q_1,\cdots,q_{i-1}] = 1-p$. 令 $J_i = \sum_{1\leq j\leq i}(q_i-(1-q))$，所以 $J_{i+1}-J_i\leq 1$。我们可以验证 J_i 序列是一个上鞅

$$E[J_i|J_1,\cdots,J_{i-1}] = E[J_{i-1}+q_i-(1-p)|J_1,\cdots,J_{i-1}]\leq J_{i-1}$$

因此通过 Azuma 不等式，

$$\Pr[J_t\geq J_1+\delta]\leq e^{-\frac{\delta}{2t}}$$

设置

$$t = \frac{k+\sqrt{k}}{p} < 16(k+\sqrt{k}), \delta=\sqrt{k}$$

我们能得到

$$\Pr[\sum_{i=1}^{t}(1-q_i)\geq k]\geq 1-e^{\frac{p}{4}}$$

这说明至少以 $1-e^{\frac{p}{4}}$ 的概率，t 时刻没有 Bad cluster。更一般地，设置 $t=O(\frac{k}{\epsilon}\log(\frac{1}{\epsilon}))$ 即允许输出的类更多的时候，我们能得到 $(4+\epsilon)$ 的近似比，比之前的 $q\cdot\log(k)$ 更好，当 k 很大的时候。