arXiv:2603.09724v1 [cs.DB] 10 Mar 2026

排名的局部稳定性

Felix S. Campbell
内盖夫本-古里安大学，以色列贝尔谢巴 felixsal@post.bgu.ac.il

Yuval Moskovitch
内盖夫本-古里安大学，以色列贝尔谢巴 yuvalmos@bgu.ac.il

摘要

排名在决策过程中发挥着至关重要的作用。然而，如果对项进行细微修改就会显著改变其排名，那么决策的质量可能会受到影响。排名稳定性是用于评估对排名算法或数据进行修改后结果所受影响的一项指标。以往的研究主要关注算法发生变化时排名的稳定性，而我们提出了一种称为 局部稳定性 的新指标。局部稳定性表示排名中某个项的数值发生细微变化时对其排名的影响。我们提出的定义还进一步考虑了排名中存在多个质量相近的项这一情况，这些项构成所谓的 密集区域，从而允许在该区域内通过细微修改交换项的位置。我们证明，一般而言，计算这一指标是困难的，因此提出了对该定义的放宽，以实现近似计算。

我们介绍了 *(i)* LStability，这是一种用于近似局部稳定性的基于抽样的算法；通过使用集中不等式，我们为其提供了概率近似正确类型的保证；以及 *(ii)* Detect-Dense-Region，这是一种基于该方法的算法，用于检测某个项所属的密集区域（如果该区域存在）。我们对算法进行了多项优化，以提升其可扩展性和效率。我们通过一系列广泛的实验验证了所提出的框架，其中包括突出展示我们定义效用的案例研究。

CCS 概念

• **信息系统** → **数据管理系统**；• **社会与专业主题** → **社会技术系统**。

关键词

排序稳定性、排名解释、敏感性分析

ACM 参考格式：

Felix S. Campbell 和 Yuval Moskovitch。2018。排名的局部稳定性。载于请填写版权确认邮件中提供的正确会议名称（会议缩写 'XX）。ACM，纽约州纽约市，美国，18页。https://doi.org/XXXXXXX.XXXXXXX

1 引言

根据定量属性对项或个人进行排名，往往在众多领域应用中发挥关键作用，例如，

允许免费制作本作品全部或部分内容的数字版或纸质副本，供个人或课堂使用，但不得出于盈利或商业目的制作或分发副本，且副本必须附有本声明及首页上的完整引文。对于由作者以外的其他方拥有的本作品组件，必须尊重其版权。注明出处后可摘录。若要以其他方式复制、再版、发布至服务器或分发至列表，须事先取得明确许可和/或支付费用。请联系 permissions@acm.org 申请许可。会议缩写 'XX，纽约州伍德斯托克 ©2018 版权所有归所有者/作者所有。出版权已授权给 ACM。ACM ISBN 978-1-4503-XXXX-X/18/06https://doi.org/XXXXXXX.XXXXXXX

表 1：根据调整后的平均发表论文数量排名的计算机科学系前十名。每组以单一颜色突出显示的行代表排名中的一个密集区域，其中各系彼此相似。

	大学	AI Pubs.	系统论文	得分 ↓
t_1	Lakefront University	44	36	39.2
t_2	Dempster University	42	35	37.9
t_3	西部理工学院	43	33	36.7
t_4	草原大学	23	25	25.4
t_5	奥格登大学	22	24	24.4
t_6	凯齐研究所	20	24	23.8
t_7	蓝岛大学	21	22	22.7
t_8	普莱恩菲尔德学院	7	13	11.9
t_9	欧文大学	8	11	11.0
t_{10}	金博尔学院	6	10	9.6

在学术界 [6, 29], 招聘 [20], 或电子商务 [15]。排名的一个主要基本假设是，排名较高意味着效用显著优于排名较低的项。然而，如果对数据进行细微修改就会导致某个项的排名位置大幅变化，那么这一基本假设便会受到削弱。举一个简单的例子，考虑一场赛车比赛的结果：如果获得第1名的车手仅以0.2秒的间隔领先，那么他与亚军之间的差距实际上可能微不足道；而领先20秒则表明其表现明显更出色。对于这些简单情形，效用的提升显而易见；但在更复杂的场景中，例如学习排序（LtR）[48]或基于多个属性计算得分的排名，如何界定各项之间的间隔并非易事，下面将对此进行说明。

例1.1。 在选择就读大学时，学生通常会参考第三方编制的潜在大学排名（例如，CSRankings [5]），以选择最能帮助自己实现目标的项目。作为贯穿全文的示例，我们将对虚构大学进行排名。为便于说明，该排名基于简化版的 CSRankings。表1展示了按调整后平均发表数量降序排列的前10 所大学，其数值由以下方式计算得出

$$\varphi(t) = \sqrt[17]{(t.\texttt{AI Pubs.}+1)^5 \cdot (t.\texttt{Systems Pubs.}+1)^{12}}$$

直观而言，该得分是人工智能和系统领域的 5 个和 12 个子领域中调整后的出版物数量的几何平均数。在这种情况下，尽管 Lakefront University 根据排序函数排名第 1，但如果系统领域的出版物仅少 4 篇（其他数据保持不变），它就会排在第 3 位。由于出版物数量可能随时间略有变化，这可能会引发诸如 “这所大学在排名中获得这一位置是否实至名归？” 或 “这所大学在排名中是否仅以微弱差距屈居第二？”之类的问题。

以及更宏观层面的问题，例如“这两所大学在排名中发生变动的可能性有多大？”

量化排名中各项位置的合理性，称为排名 稳定性，此前已有研究探讨过这一问题 [2, 3, 10, 16, 21, 46]。具体而言，在 [3], 中，排名稳定性用于衡量排名对于生成排名所用排序函数变化的稳健性。该定义给出了全局 排名稳定性得分，并揭示了排名相对于排序函数不确定性的稳定程度。

排名中经常会出现各方面相近的项，例如地位相当的大学，这可能在排名中形成密集区域。在这些区域内，细微变化就可能合理地导致各项之间交换位置。例如，美国国家研究委员会对博士项目的排名就注意到了这一现象 [29]，并为每所大学提供可能的排名范围，以体现所收集数据中的不确定性。

例*1.2*。 表1 中的大学分为三个密集区域。每个密集区域都对应一组由给定排序函数生成的分数相近的大学。如例1.1所示，只需对出版物数量进行小幅（假设性）调整（总共4篇），排名第1的大学就会变为第3名。相比之下，要让 Lakefront University 升至第4名，出版物数量需要变动17篇。排名中存在密集区域可能会影响决策。例如，设想一名学生被排名第4至第7的大学录取。虽然首选应当是排名最高的选项，但由于这些大学都属于同一个密集区域，其质量可能会被视为大致相当。因此，地理位置或项目匹配度等其他因素，可能会优先于精确的排名顺序。

[3] 中提出的排序稳定性度量具有粗粒度特性，对排名中的所有变化一视同仁——例如，一对项的一次对换与排名的完全逆转同等重要。因此，该度量可能会忽略数据中的密集区域。这促使我们提出局部稳定性的概念，即将稳定性视为排名中各个项的性质，而非整个排名的性质。将稳定性视为单个项的性质，能够更自然地处理密集区域，而密集区域本身就是排名中局部区域的性质。我们的贡献可概括如下。

局部稳定性。 我们定义了一种局部稳定性度量，旨在通过刻画能够显著改变某个给定元组位置的变更（或细化）幅度，量化数据修改对排名结果的影响。为了考量数据修改对排名的影响，我们借助反事实推理的概念 [28, 37, 42, 43]。直观而言，我们对数据中的单个元组进行小幅假设性（反事实）修改，以确定该元组相对于排名中其他元组的间隔。将关注范围限制在单个元组上，可得到一种局部稳定性度量，反映了最佳情形分析：假设数据库中其余元组均不存在不确定性；否则，在许多情况下，不确定性会加剧不稳定性。直观而言，能够显著改变元组位置的修改幅度越小，则局部稳定性越低。我们的定义取决于用户对何为位置显著变化的界定，具体通过选择一个值 k 来表示元组原始排名附近的位置范围。随后可利用此参数 k 在评估局部稳定性时考虑密集区域。此外，我们的定义还纳入合理变更集合，该集合可依据领域专门知识确定，也可采用数据驱动的方式确定，并对所考虑细化的幅度设定上限。由于局部稳定性关注的是数据中的变更，我们可以将排序函数视为黑箱，从而使定义适用于任何排序函数，包括由复杂的 LtR 模型得出的排序函数。

简而言之，细化可按对每个属性值所施加修改的幅度进行偏序。例如，延续例1.1，将人工智能发表成果的数量改变3、将系统出版物的数量改变2，被视为比仅将人工智能发表成果的数量改变2更大的修改。利用这一偏序，我们可以定义一个稳定区域，直观而言，对给定元组的值所做的任何扰动都不会使其在排名中的位置变化超过给定值 k。局部稳定性值随后被定义为稳定区域相对于用户所给合理变化整体空间的相对比例。

虽然很有用且信息丰富，但计算元组的局部稳定性是不可行的。为克服这一挑战，我们提出一种宽松的定义，允许稳定区域可能包含少量细化，这些细化会使元组在排名中的位置变动超过 k（我们将此类细化称为 k-不稳定细化），同时仍将从稳定区域中均匀抽样到此类细化的概率保持在较低水平。

估计局部稳定性。 我们提出一种两阶段的基于抽样的算法：首先从合理变化空间中抽样，以构造一个（近似的）稳定区域；随后，在该区域内对细化进行抽样，验证生成的稳定区域是否近似稳定。我们证明，只要抽取足够多的样本，就能以很高的概率验证计算出的稳定区域是否确实近似稳定（即是否仅包含少量不稳定细化）。我们提出三种优化方法，以提升基于采样的估计器的性能。

检测密集区域。 元组的局部稳定性用于评估其与在排名中所分配位置的契合程度，并可用于回答诸如“这所大学在排名中是否配得上其位置？”之类的问题，如例1.1所示。另一个相关的重要问题是确定元组在排名中的位置可能变动的范围。为此，我们提出`Detect-Dense-Region`，其目标是针对给定元组，确定一个值 k，该值涵盖该元组密集区域的范围。我们借助局部稳定性的定义，判断反事实场景中元组之间的间隔何时足够小，从而表明密集区域的存在及其范围。

实验评估。 最后，我们通过使用真实数据和

表2：符号说明

符号	说明
$\varepsilon(t)$	通过用 ε细化 t得到的元组
$D_{t\to t'}$	数据库 D，其中元组 t被替换为 $t_\prime$
$\Delta_{f(D)}(t,\ t')$	将 t替换为 $t_\prime$时的位置变化 在 $f(D_{t\to t'})$下
$\preceq$	细化包含关系
$\mathsf{E}(\cdot)$	非包含细化集合
RC	合理变更集合
k-$\mathrm{SB}^{t}_{f(D)}$	k-稳定区域边界
α-k-$\mathrm{SB}^{t}_{f(D)}$	α-k-稳定区域边界

合成数据进行广泛的实验评估，验证了我们的定义以及所提出的算法 LStability 和 Detect-Dense-Region，展现了这一方法的实用性和算法的效率。我们介绍了两个案例研究，说明我们的定义能帮助从真实数据的排名中获得洞见。例如，我们发现，在 NBA 球员排名中，学习型排序函数对乔尔·恩比德过拟合，说明他的高排名缺乏依据。此外，我们发现，[5] 对于大多数排名前十的大学，CSRankings 排名具有相当高的局部稳定性，这支持了其可靠性。实验结果还表明，`Detect-Dense-Region` 能够准确识别排名中观察到的稠密区域。我们通过比较局部稳定性度量与文献 [3], 中定义的全局稳定性度量，结束实验评估，并展示这两种定义可能会对排名产生截然不同的解读。

论文结构。 本文其余部分的组织如下。在 第2节中，我们介绍必要的背景和形式化定义，然后定义排名的精确局部稳定性和近似局部稳定性。在 第3节中，我们提出近似计算排名中某个项的局部稳定性的技术，并在 第4节中提出对算法的优化。我们在 第5节中介绍一种启发式方法，用于检测某个项周围的密集区域。在 第6节中，我们提供案例研究和对方法性能的实证分析，并在第7节中综述相关工作。最后，我们在 第8节中总结全文并讨论未来研究方向。

2 问题表述

局部稳定性度量小幅修改元组对其在排名中位置的影响，同时也考虑密集区域：在这些区域中，多个元组的质量相近，细微变化可能合理地导致元组之间交换位置。我们假设数据修改可以独立进行，也就是说，更改一个元组时，无需修改其他任何元组即可保留数据语义。在此假设下，我们关注对单个元组的修改。我们将排名 视为元组集合 [30] 的排列。排序函数 是一个函数 f，它将数据库 D 映射为其中元组的一个排列。我们用 $f(D)$ 表示在数据库 D上应用 f后得到的排名，并用 $f(D)[t]$ 表示 t 在排名中的位置。对于任意数据库 D，我们只考虑其数值属性。这一假设是标准做法，也符合先前的研究（例如，[3, 11, 30]）。

2.1 细化

我们的目标是了解元组不同属性的值发生变化时，会如何影响它们在排名中的位置。为此，我们首先定义元组的细化，用于描述对元组可能进行的修改及其对元组值的影响。

定义 *2.1*（细化）。 给定一个数据库 D及其数值属性 $a_1,\ldots,a_n$，一个细化 是一个向量 $\varepsilon=\varepsilon_1,\ldots,\varepsilon_n\in\mathbb{R}^n$。给定一个细化 ε，细化元组 $\varepsilon(t)$ 是元组 $t_\prime$，使得对于所有 $i\in[n]$，

$$t'.a_i = t.a_i + \varepsilon_i$$

例 *2.2*。 接着看例1.1。考虑细化 $\varepsilon=\langle -10,\ 5\rangle$，我们得到 $\varepsilon(t_1)$ 是元组$(34,\ 41)$。

为了理解元组属性的变化如何影响其在排名中的位置，我们定义两个元组之间的位置变化概念。

定义 *2.3*（位置变化）。对于数据库 D 和元组 t，令$D_{t\to t'}$为将 t 替换为 $t_\prime$后得到的数据库

$$D_{t\to t'} = D\setminus\{t\}\cup\{t'\}$$

用 $t_\prime$ 替换 t 所产生的位置变化为

$$\Delta_{f(D)}(t,t') = |f(D)[t] - f(D_{t\to t'})[t']|$$

例 *2.4*。 考虑细化 $\varepsilon=-10,\ -5$。细化元组 $\varepsilon(t_1)=(34,\ 31)$的得分为32.9，低于 t_3的得分。因此，我们有$\Delta_{f(D)}(t_1,\ \varepsilon(t_1))=2$，因为 $\varepsilon(t_1)$在$D_{t\to\varepsilon(t_1)}$中排名第3。

回想一下，我们的目标是确定对元组进行微小修改会如何影响其在排名中的位置，同时考虑密集区域。为此，我们引入一个参数k，它定义了元组周围的一段位置范围，我们希望据此评估其局部稳定性[1]。随后可以用这个参数来表示元组周围区域的大小（即排名高于或低于该元组至多 k 个位置的元组），这些元组的质量近似相同，也就是说，这就是密集区域。我们假设 k作为输入给定，不过我们会在第 5 节提出一种启发式方法，用于确定一个合适的 k 值，以便捕捉元组周围的密集区域（如果存在）。现在我们可以定义k-(不)稳定细化。

定义 *2.5*（k-(不)稳定细化）。给定一个数据库 D、一个排序函数 f、一个细化 ε、一个值 k 和一个元组 $t\in D$，当且仅当 $\Delta_{f(D)}(t,\ \varepsilon(t))\le k$ 时，我们称 $\varepsilon(t)$ 对于 t 在 $f(D)$ 上是 k-稳定的。否则，我们称 $\varepsilon(t)$ 是 k-不稳定的。

例 *2.6.* 再次考虑细化 $\varepsilon=\langle -10,\ -5\rangle$，如 例 2.4 中所示。由于 $\Delta_{f(D)}(t_1,\ \varepsilon(t_1))=2$，因此 $\varepsilon(t_1)$ 在 $f(D)$ 上是 1-不稳定的 t_1。

[1]在本文中，为使表述清晰，我们讨论范围沿排名两个方向同等延伸的对称情形。将我们的定义和方法加以扩展，以使这些范围不对称，是直截了当的。

2.2 局部稳定性

为了量化对元组进行小幅修改对其排名位置的影响，我们使用实现这一变化所需改动的幅度。由于小幅改动比大幅改动更有可能发生，我们主要关注导致某个项在排名中移动超过 k 个位置的最小改动。例如，在我们的示例中，拥有 ±2 个系统出版物比拥有 ±10 个系统出版物更有可能。基于这一点，我们首先在细化之间建立一个偏序。

定义 *2.7*（细化包含）。我们称细化 ε 包含于细化 ε'（且 ε' 包含 ε），并将其记为 $\varepsilon \preceq \varepsilon'$，如果对每个 $i \in [n]$，$|\varepsilon_i| \leq |\varepsilon'_i|$。

细化包含是相对于绝对值定义的，因为我们主要关注所作细化的幅度。例如，细化 $\varepsilon' = 10$，6包含细化 $\varepsilon = 10$，-5，因为 $|-5| \leq |6|$。自然地，当 $\varepsilon \preceq \varepsilon'$ 且存在 i 使得 $|\varepsilon_i| < |\varepsilon'_i|$ 时，我们记 $\varepsilon \prec \varepsilon'$，即，它是严格细化包含关系。现在我们可以如下定义元组 t 的一个 k-稳定区域边界。

定义 *2.8*（稳定区域边界）。给定一个数据库 D、一个元组 $t \in D$、一个排序函数 f 和一个值 k，令 U 为满足 $\varepsilon(t)$ 是 k-不稳定的细化集合，并令 U^+ 为 U 在 $\mathbb{R}^n_{\geq 0}$ 上的投影。那么，t 的 k-稳定区域边界是集合

$$k\text{-SB}^t_{f(D)} = Min_{\prec}(U^+)$$

其中

$$Min_{\prec}(R) = \{\varepsilon \in R \mid \nexists \varepsilon' \in R,\ \varepsilon' \prec \varepsilon\}$$

直观而言，稳定区域边界由 k-不稳定细化中幅度最小的细化构成，也就是说，不存在 k-不稳定细化包含边界中的任何细化（这也称为天空线 [7]）。

例 *2.9*。考虑一个数据集 D，它有两个属性，且其排序函数为 f，定义在 D中的元组上。图1a展示了元组周围的细化空间 $t \in D$，其中每条轴对应于 t 在各个属性上的取值。绿色区域中的细化是 k-稳定的，而红色区域由 k-(不)稳定细化构成（也就是说，集合 U中的细化）。图1b展示了细化在 $\mathbb{R}^n_{\geq 0}$上的投影。在此示意图中，绿色区域中的所有细化仍然是 k-稳定的，而较浅的红色交叉影线区域则对应于这样一些细化幅度：在投影空间中同时存在 k-稳定和 k-不稳定细化。因此，稳定区域边界 $k\text{-SB}^t_{f(D)}$ 是分隔绿色区域和红色区域的弯曲黑线。

定义 *2.10*（稳定区域）。对于一组细化 R，我们将 $\mathrm{E}(R)$ 记为不包含 R中任何细化的细化集合，即

$$\mathrm{E}(R) = \{\varepsilon \in \mathbb{R}^n \mid \nexists \varepsilon' \in R,\ \varepsilon' \preceq \varepsilon\}$$

我们将 $\mathrm{E}(k\text{-SB}^t_{f(D)})$ 称为稳定区域。

根据定义 2.8，稳定区域中的任何细化都保证是 k-稳定的。继续示例 2.9，图1b 中的绿色区域就是稳定区域 $\mathrm{E}(k\text{-SB}^t_{f(D)})$。



（a）（不）稳定细化 （b）k-$\text{SB}^t_{f(D)}$ & 合理变更

图 1：第2 节中定义的示例图示

元组的局部稳定性是相对于用户指定的一组合理变动来定义的；这组变动将可能的细化限制为对数据的微小变动，而用户认为这些变动不应导致元组排名发生显著变化（无论是利用领域专门知识，还是像我们在第 6.2 节中那样采用数据驱动的方式）。更正式地说，我们将 ε^{max}定义为向量 $\langle \varepsilon_1^{max}, \ldots, \varepsilon_n^{max} \rangle \in \mathbb{R}^n_{\geq 0}$，并且只考虑满足 ε 的细化 $\varepsilon \preceq \varepsilon^{max}$。

例*2.11*。 在例1.1的背景下，合理变化的集合可能取决于研究领域中的活跃程度。例如，如果我们假设在这些大学中，人工智能比系统更活跃，那么可以假设人工智能论文数量的方差大于系统论文数量的方差。因此，我们可以选择例如，$\varepsilon^{max} = \langle 5, 3 \rangle$，表示每所大学可能有 ±5 篇人工智能发表成果和 ±3 篇系统出版物。

我们令RC 表示包含于向量 ε^{max} （即，所有 $\varepsilon \preceq \varepsilon^{max}$）中的细化集合。例如，在图1中，RC 是这样一个细化集合：其投影到 $\mathbb{R}^n_{\geq 0}$ 后位于虚灰线界定的区域内。现在，我们已经可以定义元组的局部稳定性。

定义*2.12*（局部稳定性）。元组 t相对于数据库 D、排序函数 f、合理变更集合RC以及参数 k 的局部稳定性定义为

$$\text{Stability}_{f(D)}(t, k \mid \text{RC}) = \frac{\text{Vol}\big(\text{RC} \cap \mathrm{E}(k\text{-SB}^t_{f(D)})\big)}{\text{Vol}(\text{RC})}$$

其中 Vol( ) 表示这些集合的体积（即 Jordan 测度）。

直观而言，局部稳定性度量衡量的是限制在合理变更集合中的稳定区域与合理变更集合之间的相对大小。

例*2.13*。 对于 ε^{max} 和 合理变更 如 图1b所示，局部稳定性是限制在 合理变更 中的绿色区域与 合理变更下方区域的比值。在这种情况下，局部稳定性会很高，因为大部分区域都是绿色的。

计算 $k\text{-SB}^t_{f(D)}$ 并非易事。事实上，我们可以证明

定理2.14。 除非 $FP = \#P$，否则不存在多项式时间算法，能够在给定数据库 D、元组 $t \in D$、排序函数 f 和数值 k 的情况下计算 $k\text{-SB}^t_{f(D)}$。

证明概要。我们通过证明以下命题来完成证明：如果 k-$\text{SB}^t_{f(D)}$ 可以在多项式时间内计算，那么 $|k\text{-SB}^t_{f(D)}|$ 也可以在多项式时间内计算。然而，我们证明了 #DNF，

这一著名的 #P 完全问题 [41]，可以在多项式时间内归约为计算 $|k\text{-SB}^t_f(D)|$。因此，如果存在能够计算 $k\text{-SB}^t_f(D)$ 的此类算法，那么FP =#P。详情见附录 A。□

此外，我们预计，即使给定稳定区域边界$k\text{-SB}^t_f(D)$，计算稳定性度量仍然难以处理，因为这与计算超体积指标的问题密切相关。超体积指标计算以原点为锚点的超矩形并集的体积，且已知该问题为 #P-困难 [8, 22]。

2.3 α-局部稳定性

鉴于局部稳定性计算的复杂性，我们定义该问题的一个近似版本。首先，我们将α-k-稳定区域边界定义为一组细化，其与某个 k-稳定边界之间的误差不超过一定范围。具体而言，在均匀抽取样本时，从 α-k-稳定区域边界下的细化集合中抽到 k-不稳定细化的概率至多为 α。

定义 *2.15*（α-k-稳定区域边界）。给定数据库 D、元组 $t \in D$、排序函数 f、值 k以及合理变更集合 RC，令 U为满足 $\varepsilon(t)$为 k-不稳定的细化集合，并令 U^+为 U在 $\mathbb{R}^n_{\geq 0}$上的投影。则，t的 α-k-稳定区域边界是集合 $\alpha\text{-}k\text{-SB}^t_f(D) \subseteq U^+$，满足

(i) $\alpha\text{-}k\text{-SB}^t_{f(D)} = Min_{\prec}(\alpha\text{-}k\text{-SB}^t_{f(D)})$

(ii) $\mathbb{P}_{\varepsilon\sim\mathcal{U}(\text{RC})}[\varepsilon(t) \text{ is } k\text{-unstable} \mid \varepsilon \in \mathsf{E}(\alpha\text{-}k\text{-SB}^t_{f(D)})] \leq \alpha$

其中 U 为均匀分布。

当 α 较小时，我们可以保证，以较高概率，对元组进行 $RC \cap \mathsf{E}(\alpha\text{-}k\text{-SB}^t_{f(D)})$中的细化，不会导致其在排名中的位置偏移超过 k。现在我们可以定义 α-局部稳定性。

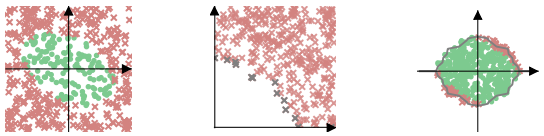
定义 *2.16*（α-局部稳定性）。相对于数据库 D、排序函数 f、$\alpha\text{-}k\text{-SB}^t_{f(D)}Sb$、合理变更集合 RC 以及参数 k，元组 t的 α-局部稳定性定义为

$$\alpha\text{-Stability}_{f(D)}(t, k \mid Sb, \text{RC}) = \frac{\text{Vol}(\text{RC} \cap \mathsf{E}(Sb))}{\text{Vol}(\text{RC})}$$

3 计算局部稳定性

我们提出了**LStability**，这是一种基于采样的算法，用于在给定数据库 D、元组 $t \in D$、排序函数 f、合理变更集合 RC、值 k以及与近似质量和置信度相关的参数时，估计 α-局部稳定性；这些参数将在下文讨论。该算法利用了以下事实：尽管计算稳定区域边界存在困难（定理2.14），但给定一组细化 R，我们可以高效地界定 α，使得R 以较高概率是一个 $\alpha\text{-}k\text{-SB}^t_{f(D)}$。我们首先说明，只要样本数量足够多，就能估计给定集合 R 包含不稳定细化 k的概率。

命题 3.1。 给定数据库 D、排序函数 f、值 k、元组 $t \in D$、置信水平 δ、置信区间 η，以及



(a) 构造 **(b)** Sb **(c) 验证**

图2：基于 LStability，以 图1所示的细化空间为基础，对其步骤的示例说明。(a) 绿色（圆形）得分表示采样得到的稳定细化，而红色（十字形）得分表示不稳定细化。(b) 将 k-不稳定细化投影到 C上的 $\mathbb{R}^n_{\geq 0}(U^+_C)$。灰色（十字形）得分表示估计稳定区域边界上的得分。(c) 从估计稳定区域 RC ∩ E(Sb) 中获取用于验证步骤的样本。

一组细化 R，令

$$p = \mathbb{P}_{\varepsilon\sim\mathcal{U}(RC\cap E(R))}[\varepsilon(t) \text{ is } k\text{-unstable}]$$

$$\widehat{p} = \frac{1}{N}\sum_{\varepsilon\in\mathcal{S}} \mathbb{1}[\varepsilon(t) \text{ is } k\text{-unstable}]$$

其中，$\mathcal{S}$ 是从 $RC \cap E(R)$中独立同分布地均匀抽取的 $N = \frac{1}{2\eta^2}\ln\frac{1}{\delta}$个样本的集合。于是有

$$\mathbb{P}[p - \widehat{p} \leq \eta] \geq 1 - \delta$$

证明概要。该结论可直接应用 Hoeffding不等式 [23] 得出。□

以下是 命题 3.1的一个推论

推论3.2。 $N = \frac{1}{2\eta^2}\ln\frac{1}{\delta}$ 个从 $RC \cap E(R)$ 中独立同分布均匀抽取的样本，足以以概率 $1-\delta$确认 R 是一个 $\alpha\text{-}k\text{-}SB^t_{f(D)}$。

LStability 建立在这一思想之上。它的输入包括一个数据库 D、一个元组$t \in D$、一个排序函数 f、一个值 k、一个合理变更集合 RC、一个置信水平 δ、一个采样预算 N、一个置信区间 η，并由两个阶段组成。第一阶段生成一个细化集合Sb，根据推论3.2，该集合以较高概率是一个 $\alpha\text{-}k\text{-SB}^t_f(D)$。第二阶段中，LStability 输出由 Sb 定义的估计稳定区域的体积与 RC 的体积之比的估计值。图 2 对 LStability 进行了示意性概述。在第一阶段，LStability 计算一个细化集合 Sb，该集合以较高概率是一个 $\alpha\text{-}k\text{-SB}^t_f(D)$。这一过程通过两个基于样本的步骤完成：集合构造和验证。在图 2a 和图 2b 所示的构造步骤中，算法按如下方式进行：

(1) 从C中均匀抽取一组 N细化样本合理变更，并将 U^+_C中的 k 不稳定细化投影 C到$\mathbb{R}^n_{\geq 0}$（见图 2a）。

(2) 计算 $Sb = Min_{\prec}(U^+_C)$（见图2b）。在验证步骤中（如图图2c所示），算法执行以下操作：(3) 它抽取一组 V，包含足够数量的细化（$N = \frac{1}{2\eta^2}\ln\frac{1}{\delta}$个样本），均匀取自合理变更 ∩E($Sb$)

(4) 它计算 k-不稳定细化在 V 中占样本总数 N 的比例，从而确定 α，使得 Sb 为 α-k-SB$^t_{f(D)}$ 的概率为 $1-\delta$，依据推论3.2

最后，在第二阶段，给定第一阶段得到的 Sb，LStabilit y $D)$ $(t, k \mid Sb,$ 合理变更）。在能够保证稳定区域为凸集的情况下（例如，当排序函数基于由属性的线性组合计算出的分数时），可以使用其他方法作为直接替代方案。具体而言，[14, 19] 介绍了用于估计凸多胞体体积且具有理论保证的高效算法。尽管在适用时，这些方法在准确性和效率方面都能提供更强的理论保证，但蒙特卡罗方法通常效率较高，且对我们的用途而言足够准确。此外，我们注意到，在许多实际场景中，凸性假设并不成立。

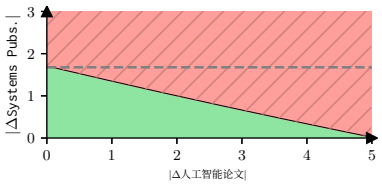


图 3：Dempster Uni. 和 $k=0$ 的细化稳定性。绿色（实线）区域中的细化是0-稳定的，红色（阴影线）区域中的细化是0-不稳定的。包含稳定区域边界上所有细化的缩减合理变化集合，由灰色虚线下方的细化组成。

4 优化

接下来，我们概述针对 LStability 的三项优化。第一项缩小合理变更集合（合理变更），使算法能够更有效地利用采样预算。第二项优化旨在降低评估排序函数 f 的成本。当修改数据中的某个元组不会影响排序函数所得其他元组的相对顺序时，该优化便适用，而这是排序函数的一个常见性质。最后，我们建议对 α 加以限制，以缩短运行时间。具体做法是迭代地 在每次迭代中使用部分采样预算运行构造步骤和验证步骤，从而在达到期望的 α 水平时允许提前终止。

4.1 缩小合理变更集合

第一项优化旨在通过缩小合理变更集合来提高采样预算的效用，同时保证该集合仍包含稳定区域边界下的所有 k稳定细化。使用更小的合理变更集合可以缩小 LStability 构造步骤的样本空间，从而实现更具信息量的抽样，因为被排除的细化无法对稳定区域边界的计算做出贡献。

为缩小合理变化集合，我们使用单维细化，即修改单个属性的细化。

考虑此类细化可以排除不可能位于稳定区域边界内的细化。更正式地，考虑单个属性 a_i，以及满足 $|\varepsilon_i| > 0$ 且对每个 $j \neq i$ 都有 $\varepsilon_j = 0$ 的细化 $\varepsilon \in \mathbb{R}^n$。此外，假设对于 $f(D)$，$\varepsilon(t)$ 是 k-不稳定的。根据k-SB$^t_{f(D)}$的定义，我们因此可以得到，任何满足 $|\varepsilon_i| \leq |\varepsilon_{i'}|$ 的细化 $\varepsilon_{i'}$ 都不可能属于 E(k-SB$^t_{f(D)}$)。

示例 *4.1.* 考虑我们一直使用的示例，以及 $\varepsilon^{max} = \langle 5, 3 \rangle$，其定义见 示例 2.11。图 3 展示了 Dempster Uni. 和 $k=0$ 的细化的稳定性。如图 3所示，考虑细化 $\varepsilon = 0, 2, \varepsilon(t_2)$ 对于 $f(D)$ 是 0-不稳定的。因此，任何包含超过 ±2 篇系统出版物的细化都不可能位于 0-SB$^t_{f(D)}$ 中。事实上，任何位于灰色区域上方的细化都不可能属于稳定区域边界。

为了减少合理变更 (RC)，我们将一部分采样预算用于对每个属性 a_i 的单维细化 ε_i^{max} 进行抽样。令 ε_i^* 表示样本中满足 $\varepsilon(t)$ 对于 $f(D)$ 是 k-不稳定的 ε_i 的最小（绝对）值。随后，我们将 RC* 定义为满足 $\varepsilon_i^{max} = \varepsilon_i^*$ 的合理变更集合。LStability 随后使用 RC* 替代 RC；给定 α-稳定性$f(D)$ $(t, k \mid Sb,$ RC$^*)$ 的估计值后，我们将其乘以因子 Vol(RC*) Vol(RC)，得到 α-稳定性$f(D)$ $(t, k \mid Sb,$ RC)。当排序函数具有单调性时，使用二分搜索可以高效地缩减 RC [18],，具体方法详见附录 B。

4.2 降低重新排序成本

要确定细化 ε 是否 k稳定，LStability 需要计算值 $\Delta_{f(D)}(t, \varepsilon(t))$，而这又取决于值 $f(D_{t\to\varepsilon(t)})[\varepsilon(t)]$。函数 f 对数据库 D 大小的依赖可能非常复杂，例如学习排序函数的情形。因此，计算 $f(D_{t\to\varepsilon(t)})$ 可能代价高昂，尤其是在采样预算 N 较大的情况下。然而，我们注意到，对于下文将要定义的元组独立排序函数，确定 $\Delta_{f(D)}(t, t') \leq k$ 无需付出计算 $f(D_{t\to\varepsilon(t)})$ 的代价。

定义 *4.2*（元组独立排序函数）。若对于任意 $t \in D$ 和通过细化 t 得到的 t'，以及所有满足 $f(D)[a] < f(D)[b]$ 的 $a, b \in D \setminus \{t, t'\}$，均有 $f(D_{t\to t'})[a] < f(D_{t\to t'})[b]$，则称排序函数具有**元组独立性**。

元组独立性是排名函数的一项常见特征。例如，基于例 1.1中属性几何平均数的评分函数具有元组独立性，因为更改一个元组的属性只会影响该元组的得分，而其余元组之间的相对顺序保持不变。

给定元组 t 和细化 ε，令 $t^{\uparrow}$ ($t^{\downarrow}$) 表示在 $f(D)$ 中位于 $k+1$ 上方（下方）的元组位置，其中 f 是元组独立排序函数。由于除细化后的元组 $\varepsilon(t)$ 之外，所有元组之间的相对顺序均保持不变，因此，要确定 $\Delta_{f(D)}(t, \varepsilon(t)) \leq k$，只需检查 $\varepsilon(t)$ 与 $t^{\uparrow}$ ($t^{\downarrow}$) 之间的相对顺序是否不同于 t 和 $t^{\uparrow}$ ($t^{\downarrow}$) 之间的相对顺序。因此，我们只需在三个元组上评估 f（数量为常数）。当数据库规模较大，且排序函数的评估高度依赖数据规模时（例如在学习排序模型中），这一点可能很重要，正如我们的实验评估（第 6.5 节）所示。

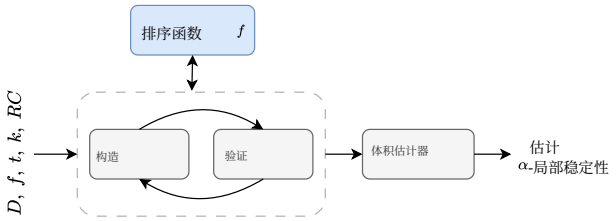
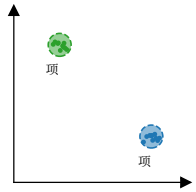


图4：LStability 组件概览

X	Y	得分 ↓
9.71	0.86	10.57
1.38	9.15	10.53
9.30	1.20	10.51
...	...	...
8.31	0.52	8.83

(a) 数据集的分数



(b) 数据集可视化

图5：示例展示了属性空间中两个相距较远的簇如何映射到相似的分数，这表明它们都属于排名中的同一个密集区域

4.3 为有界 α减少采样

LStability 返回其计算出的细化边界所达到的极小性级别 α。不过，选择一个期望的 α 是很自然的；为了获得一组细化 Sb，使得 Sb 是具有期望 α 的 α-k-SB$^t_f(D)$，构造步骤可能没有必要像实际那样进行那么多次采样。为此，我们扩展了 LStability，使其能够如图 4 所示，迭代执行构造步骤和验证步骤。在每次迭代中，算法使用有限数量的样本计算一个集合 Sb，随后检查对于给定界限 $\alpha \leq$，它是否为 α-k-SB$^t_f(D)$。如果是，算法便继续进行 Sb 和 RC 的体积估计。否则，下一次构造步骤会将上一次验证步骤中不稳定的细化合并到 Sb 中，利用这些反例改进其估计，并且只从 RC $\cap$ E(Sb) 中采样，以确保采样得到的细化只能改进对 Sb 的估计。该过程运行 ℓ 次迭代，具体次数由用户确定。

构造步骤在每次迭代中所抽取的样本数量会带来权衡。目标是仅抽取足够数量的样本，以获得一个足够好的估计稳定区域边界，即一个满足所需 α的 α-k-SB^t_f（D）集合。然而，当需要多次迭代才能获得一个集合 Sb，使其针对给定的 α满足 α-k-SB^t_f（D）时，验证步骤会被多次执行。我们建议将预算平均分配给每次迭代，同时考虑验证步骤在每次迭代中抽取的样本。具体而言，如果在 LStability 原始版本中，分配给构造步骤的样本数为 N，分配给验证步骤的样本数为 V，则我们在每次迭代中为构造步骤分配$\frac{N+V}{\ell} - V$个样本。在这种分配方式下，即使在最坏情况下，我们抽取的样本数也与原先预算的数量相同。

5 检测密集区域

元组 t 的局部稳定性是相对于给定的 k定义的，它规定了评估局部稳定性的位置范围。这种参数化使我们对局部稳定性的定义在排名中存在密集区域时仍然适用。到目前为止，我们一直关注如何计算排名中给定元组的局部稳定性。在我们的贯穿示例中，局部稳定性
该度量评估某所特定大学与其在排名中所处位置的契合程度。一个有趣的替代视角是：对于给定的大学，根据局部稳定性度量，探究其在排名中可能位置的界限。也就是说，识别排名中某个元组周围的密集区域（如果该元组位于密集区域中）。

算法1： Detect-Dense-Region

输入： 数据库 D、一个元组 $t \in D$、一个排序函数 f，以及一组合理变化 RC 和采样预算 N
输出： 一个值 k（t 的密集区域大小）

$S \leftarrow \emptyset$
对于 $i \leftarrow 1$ **到** N **执行**
  $\varepsilon \leftarrow$ 样本(U (合理变更))
  $t' \leftarrow \varepsilon(t)$
  $S \leftarrow S \cup \{(\varepsilon, \Delta_{f(D)}(t, t'))\}$
$k^* \leftarrow$ 最大值($\{p \mid (\varepsilon, p) \in S\}$) ▶ 位置的最大变化由合理变更的样本诱导
稳定性$(t, k^*) \leftarrow 1$
对于 $k \leftarrow 0$ **到** $k^* - 1$ **执行**
  $S \leftarrow S \setminus \{(\varepsilon, p) \in S \mid p \leq k \wedge (\nexists(\varepsilon', p') \in S, (p' > k) \wedge (\varepsilon' \leq \varepsilon))\}$
  稳定性$(t, k) \leftarrow 1 - \frac{|S|}{N}$
$d_0 \leftarrow$ 稳定性$(t, 0)$
对于 $k \leftarrow 1$ **到** k^* **执行**
  $d_k \leftarrow$ 稳定性(t, k) − 稳定性$(t, k-1)$
$C_s, C_l \leftarrow$ 聚类$(d_0, \ldots, d_k)$ ▶聚类 返回一个划分，对象为 $d_0, \ldots, d_k$ 分为两组：小和大
返回 最小值$_{k \in [k^*],\ d_k \in C_l}\ k$

直观而言，为了识别密集区域，我们需要将排名中效用相似的项进行聚类。然而，当排名不是基于分数的（例如，某些列表式学习排序模型 [25]）时，这可能无法实现，因为此时无法为给定项定义有意义的效用。另一种朴素的解决方案是根据元组的属性进行聚类。然而，这种方法假设排名相近的所有元组在属性上也必须相似，但情况并不总是如此，正如下面将要展示的那样。

示例 *5.1*。 图 5a 展示了由两个属性组成的项 x和 y按 $x + y$降序排列，每项在 图 5b中以一个点表示。如 图 5b所示，根据图中以蓝色和绿色虚线圆圈表示的属性值，可以观察到两个簇。然而，如 图 5a所示，它们的分数都相近，表明所有项都属于同一个密集区域。

为此，我们提出**Detect-Dense-Region**，一种启发式方法，用于为数据库中的给定元组计算值 k，并遵循透过该视角所看待的密集区域的直观含义

我们的局部稳定性定义。其高层思想是Detect-Dense-Region在于比较给定元组在不同 k 取值下的稳定性值差异，并找出出现较大差异之前的最后一个 k 值。以下为Detect-Dense-Region 的伪代码，如算法1所示。该算法首先估计不同 k 值下的局部稳定性（第1-10行）。然后（第11-15行），该算法使用聚类将计算出的稳定性值差异划分为较小变化和较大变化，并返回所建议的 k，即首次观察到较大差异的位置。接下来，我们详细介绍各个阶段。

针对多个取值估计局部稳定性 k。为了确定我们观察到估计局部稳定性出现较大差异的第一个 k 取值，我们首先需要估计 k不同取值下的局部稳定性。这里的挑战在于如何高效地完成这一过程。当我们只需要局部稳定性的粗略估计时，对 k 的所有可能取值评估 LStability 会带来不必要的高昂开销。

为此，Detect-Dense-Region 根据给定的采样预算 N 对细化进行抽样，并将每个抽样得到的细化 ε连同位置变化 $\Delta_f(D)(t, \varepsilon(t))$ 存储在样本集合 S中（第1–5行）。抽样之后，算法确定由任意样本引起的最大位置变化，记为 k^*（第6行）。这些样本用于估计 0 到 k^* 之间每个 k 的局部稳定性，记为 $\widehat{\text{Stability}}(t, k)$。$k^*$ 被假定为合理变更集合 RC 中任意细化所引起的最大位置变化（当样本数量足够大时，这一假定成立），因此 k^* 的估计稳定性为 1（第7行）。对于从 0 到 $k^* - 1$ 的每个 k 取值，均使用样本集合 S估计稳定性。直观而言，给定 k的稳定性值可估计为：S 中满足 ε为 k-稳定（即 $p \leq k$）、且不存在 $(\varepsilon' p') \in S$ 使得$\varepsilon' \leq \varepsilon$ 且 ε' 为 k-不稳定（即 $p' > k$）的样本 (ε, p) 数量，除以样本总数 N。请注意，估计的 $(k+1)$-稳定区域边界高于估计的 k-稳定区域边界。因此，估计可以按如下方式迭代进行（第8–10行）：对于每个 k，我们从 S 中移除不包含任何已抽样 k-不稳定细化的 k-稳定细化（第9行），并将 k 的局部稳定性估计为 $1 - \frac{|S|}{N}$（第10行）。

—

基于聚类的密集区域建议。给定 k观测值的局部稳定性估计，根据我们对密集区域的定义，元组所在密集区域的范围，是 k 的第一个取值；在该值处，后续的 k 具有显著更高的局部稳定性。因此，Detect-Dense-Region会计算每个 k、 d_k取值对应的稳定性差值（第11—13行），即 k的局部稳定性减去 $k-1$ 的局部稳定性（并固定 $d_0 = \widehat{\text{Stability}}(t, 0)$）。然后，它采用Fisher-Jenks自然断点法 [17] 聚类，将这些差值划分为两组：分别是小差值 C_s 和大差值 C_l（第14行）。最后，Detect-Dense-Region返回 k，其中 k 是满足 d_k 属于大差值集合 C_l的最小取值（第15行）。

6 实验

本节将介绍实验评估。首先，我们通过两个案例研究来展示局部稳定性定义的实用性。随后，我们研究采样预算 N、置信水平 δ 和置信区间 η 对 LStability 性能的影响。我们使用真实数据和合成数据考察 Detect-Dense-Region 的性能，并展示该算法的可扩展性以及所提出的优化的实用性。最后，我们将其与 [3] 中提出的稳定性定义进行比较。

6.1 实验设置

实现细节。实验在 Google Cloud Compute Engine c4d-highcpu-4 服务器上进行，该服务器配备 4 个 vCPU（AMD EPYC 9005 系列）和 7 GB 内存。我们使用 Python3.13 实现算法，并使用 Polars[2] 1.29 加载和处理数据。每项实验重复 10 次，并在图中展示感兴趣指标的平均值（例如，运行时间、估计的 (α-)局部稳定性）。

对于 LStability 的验证步骤和优化构造步骤，我们通过拒绝采样实现对估计边界 Sb下细化项的均匀采样。具体而言，我们从合理变更中均匀采样，并拒绝包含 Sb的任何细化项的细化项。当我们要采样区域的体积相较于合理变更较小时，拒绝采样的效率可能较低。因此，我们对估计体积 τ_v设置阈值：如果在构造步骤结束时，该区域体积与其所属超矩形体积的估计比值低于 τ_v，我们就跳过验证步骤并提前停止。我们采用 $\tau_v = 0.05$，因为将估计精度提升到这一点以下，相对于验证成本而言，收益会逐渐降低。我们总共评估了 10,560 项实验，其中 7.6% 因 τ_v 而跳过验证步骤。

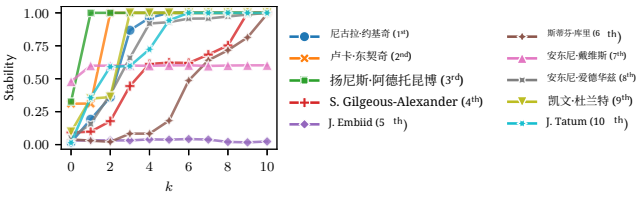
我们评估LStability在启用第 4 节 中介绍的所有优化后LStability，并将其性能与第3节中所述的未进行优化的 Basic 版本进行比较。

参数设置。 对于构造步骤，我们将样本总数设为最多 750,455 个；每次迭代抽取 20,000 个样本，最大迭代次数 ℓ 为20。我们根据第6.3节中的采样预算参数研究，通过经验选定了这个数值。对于验证步骤，除非另有说明，我们将 α 的上限设为0。05，并将置信界设为 $\delta = 0.05$。也就是说，对于给定的 k，如果我们有 95% 的把握认为估计边界内包含的 95% 细化结果都是 k-稳定的，就认为满足要求。为了有较大机会达到 $\alpha \leq 0.05$，我们将置信区间 $\eta = 0.01$设为该值。

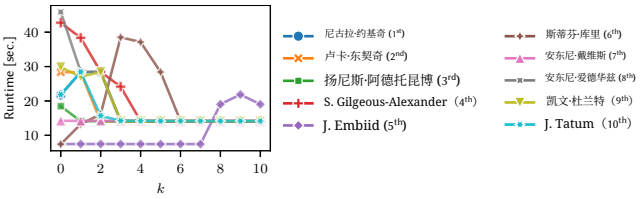
数据集与排名函数。 我们使用真实数据集和合成数据集评估了算法，并采用了以下不同类型的排序函数。

NBA（2023-2024）： 我们考虑某知名体育网站 [35] 发布的前100名球员排名。由于该排名的制定方法未公开，我们以 2023-2024 赛季的球员统计数据（从 [32] 收集）作为输入特征，根据参考排名学习一个排序函数。我们使用 LightGBM [26]，基于与 [11] 中相同的五项统计数据训练一个回归器：得分（PTS）、总篮板（TRB）、助攻（AST）、抢断（STL）和盖帽（BLK）。对于这五项统计数据，我们采用赛季总数

[2] https://pola.rs



（a） k 对稳定性值的影响



（b） k 对运行时间的影响

图6：NBA前十名球员的案例研究结果

因为我们关注的是球员整个赛季的表现排名。默认情况下，我们将合理变化（RC）的范围设为各属性最大差值的约 5%。

CSRankings：我们使用来自 [30]（基于 [5]）的数据集和排序函数，其中包含188所大学，以及这些大学在计算机科学四个不同领域的教职人员和出版物数量：人工智能（人工智能）、系统（系统）、理论（理论）和跨学科研究（跨学科）。出版物数量会根据每篇出版物的作者人数进行调整，即将份额平均分配；例如，一篇有 2 名来自不同机构的作者的论文，计为每个相关机构 0.5 篇出版物（也就是说，某个机构的出版物数量发生分数变化是合理且有意义的）。大学按照得分降序排名，得分通过对各领域调整后的出版物数量应用几何平均数的一种变体计算得出，具体如下：

$$\varphi(t) = \sqrt[27]{(t.\mathtt{AI}+1)^5 \cdot (t.\mathtt{Sys}+1)^{12} \cdot (t.\mathtt{Thry}+1)^3 \cdot (t.\mathtt{Intdsc}+1)^7}$$

我们将合理变更（RC）设为各属性最大差异的约5%——约为人工智能领域 ±4 调整后的出版物数量，以及其他领域 ±1 调整后的出版物数量。

合成数据： 我们专门针对包含密集区域的情况生成合成数据集。为了生成包含N 个元组和 d 个属性的合成数据集，我们反复生成元组数量随机的密集区域，直到数据集中包含 N 个元组。每个密集区域都会被赋予一个得分 s，与前一个密集区域之间相隔一个固定间隔 c。接着，我们从多元高斯分布中抽取元组，其均值为由 $\frac{s}{d}$缩放的全1向量，协方差矩阵则为缩放后的单位矩阵。我们根据排序函数对生成的数据集进行排序，按照元组属性之和降序排列。我们令 RC 为一个向量，其各分量与 $\frac{c}{d}$成比例，从而确保在间隔 c固定时，不同属性数量下的局部稳定性值具有可比性。

6.2 案例研究

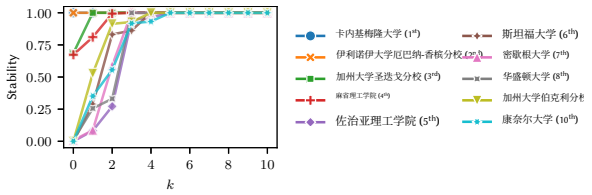
我们首先通过 NBA 和 CSRankings 数据集开展两项案例研究。

6.2.1 案例研究：*NBA*球员排名。排名在体育领域中十分常见，并能在多种情境下影响决策，例如梦幻联赛、选秀以及评估某支球队的实力。在本案例研究中，我们考察根据学习得到的排名函数评定的 2023-2024赛季 NBA 球员的局部稳定性，目标是按照球员在该赛季的表现对其进行排名。由于排名靠前的球员会受到显著更多的关注， [15, 40], 我们聚焦于排名前十名球员的局部稳定性。不同 k 取值下前十名球员的局部稳定性见图6a。

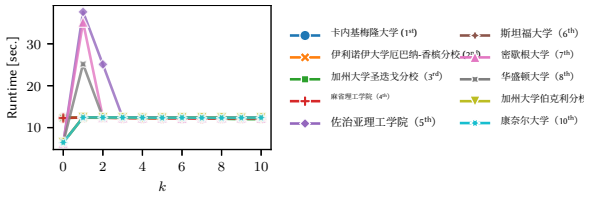
假设我们要选出本赛季最有价值球员（MVP）。根据排名，自然会选择本赛季排名第一的球员尼古拉·约基奇。然而，图6a显示，他排名第一极不稳定：当k = 0时，局部稳定性仅为0。02。事实上，只要对他该赛季的总数据稍作调整，例如， +10.04 得分、−3.04 篮板、 −0.56 助攻、 −1.01 抢断和 −1.11 盖帽，他就会跌至第二名。值得注意的是，与他整个赛季的总数据相比，这些变化尤其微小：该赛季，约基奇得到2,085分、976个篮板、708次助攻、108次抢断和68次盖帽。此外，他在前三名中的排名也不稳定；当 k = 2时，局部稳定性为0。36。这表明，依据这个排序函数，将约基奇评为本赛季 MVP 可能缺乏充分依据。根据这个排序函数，我们或许可以得出另一结论：选择卢卡·东契奇更站得住脚，因为他肯定在前四名之列（比约基奇更有可能）。或者，我们可以改进排序函数（例如，只考虑部分属性），以获得局部稳定性值更高的排名，从而为所选 MVP 提供依据。

我们观察到，大多数前十名球员在各自的位置上并不稳定：当 $k = 0$ 时，最高的估计稳定性为 0.46，出现在安东尼·戴维斯的案例中。虽然对于 $k = 0$，大多数球员都不稳定，但我们观察到，大多数球员的排名与其初始排名相差不超过 ±3 位。因此，我们可以推断，这个特定的排序函数对于前十名球员而言具有合理的局部稳定性，并确认它整体上是有根基的：也就是说，即使本赛季的统计数据发生微小变化，排名靠前的球员仍能保持高排名。一个显著的例外是乔尔·恩比德，对于所有评估的 k 值，他的稳定性都非常低。这表明学习型排序函数对恩比德的统计数据过拟合了，因为即使对他的统计数据做出极小的调整，他也会跌出前十名。恩比德在 2023-2024赛季 出战的比赛远少于其他球员（仅 39 场，而大多数其他球员超过 70 场），原因是伤病；这一事实也支持了上述解读。因此，他本赛季的总数据远低于原本可能达到的水平，使其更接近排名较低的球员。

图6b 展示了（优化后的）LStability 在不同 k 取值下、针对每位前10 名球员的运行时间。总体而言，运行时间随 k取值的增大而减少。直观上，这是因为必然存在更多 k-不稳定细化，对应于较低的 k取值，这往往导致LStability 中进行更多迭代，以产生足够好的估计。值得注意的例外是斯蒂芬·库里和乔尔·恩比德，他们的运行时间起初较低（因为他们低于估计体积阈值 τ_v），在刚刚超过 τ_v时上升，随后在其局部稳定性提高后再次下降。与基本

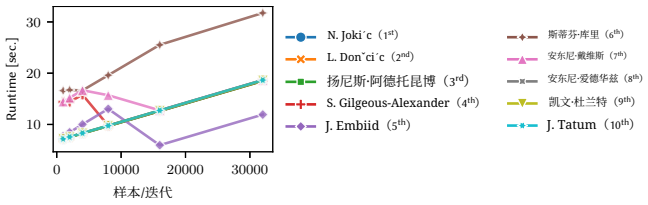


（a） k 对稳定性值的影响

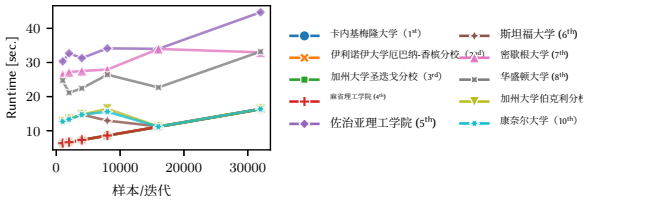


（b） k 对运行时间的影响

图7：排名前十的大学的案例研究结果

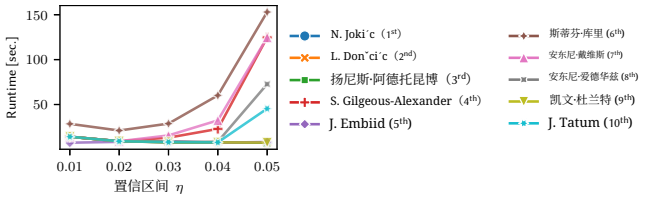


(a) NBA

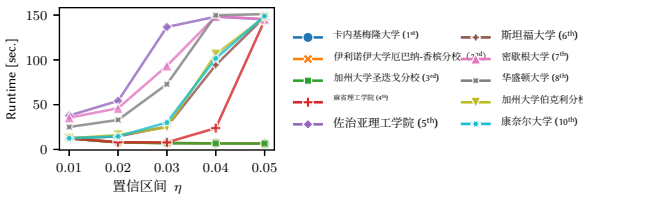


(b) CSRankings

图8：每轮构造步骤所取样本数量对运行时间的影响



(a) NBA



(b) CSRankings

图9：置信区间 η 对运行时间的影响

版本的LStability（如第3节所述）相比，我们的优化算法在最佳情况下 51.6× 更快，平均 25.4× 更快。完整结果见图18 （附录C）。

6.2.2 案例研究：*CSRankings*。排名在高等教育中同样占据突出地位，并且一直是大量研究的对象，旨在开发能够准确反映各机构相互比较时质量认知的方法 [6, 29]。我们考虑评估一种更为专门的排名CSRankings [5], 中各机构的局部稳定性，该排名依据计算机科学会议的发表指标对机构进行排名。鉴于人们对排名中靠前位置的关注，我们再次考察排名前十的大学的局部稳定性。

图7a表明该排名具有相当高的局部稳定性。事实上，排名前2的大学，卡内基梅隆大学（卡内基梅隆大学）和伊利诺伊大学厄巴纳-香槟分校（伊利诺伊大学厄巴纳-香槟分校），被估计为完全局部稳定：来自合理变更集合的任何变更都不会使它们分别排到第1名或第2名之外。对于其余大学，我们看到，对于 $k = 3$，所有大学的估计局部稳定性均超过12。此外，根据 $k \geq 5$，所有大学都被估计为完全局部稳定。这为这些大学拥有排名最高的计算机科学系这一说法提供了佐证，因为排名中的微小变化不会导致其位次发生剧烈变化。

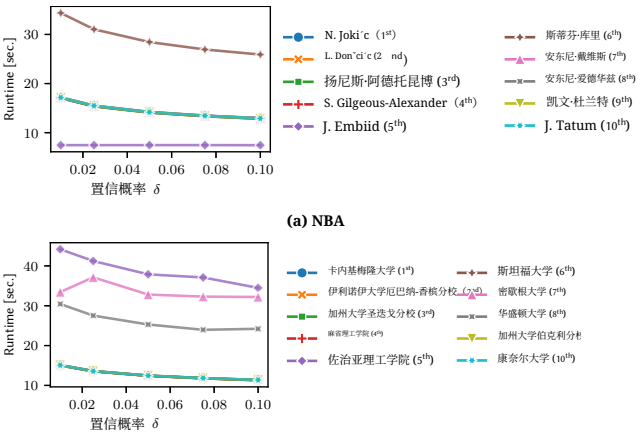
图7b 展示了（优化后的）LStability 随 k取值变化时的运行时间。正如预期，运行时间通常会随着 k的增加而缩短。与 NBA 案例研究中一样，一些大学的估计稳定性略高于阈值 τ_o，导致在这些 k取值下运行时间较长。与 35.2× 相比， 19.1× 基本版LStability，最佳情况下提速可达，平均提速。完整结果见附录C中的图18。

6.3 参数的影响

下一组实验研究采样预算N以及验证参数 δ（置信水平）和 η（置信区间）对运行时间的影响 LStability 和计算得到的稳定性值。参数对稳定性值的影响微乎其微，因此仅在 附录C中展示。对于每个数据集，我们设置 k，使不同元组中呈现出范围广泛的局部稳定性值。具体而言，对于 NBA，我们选择了 $k = 5$，而对于 $k = 1$ CSRankings，则选择了。

采样预算。 图8 展示了采样预算从1,000增加到32,000时的运行时间 N。我们观察到，每轮迭代的样本预算较小时和较大时，运行时间之间存在权衡。当单次迭代中的样本数量不足以达到预期 α时，就需要进行多次迭代，从而产生多次执行验证步骤的开销。对于局部稳定性较低的元组（例如，图8b中的康奈尔大学）。

验证参数（η， δ）。 为评估 η的影响，我们将其值从 0。01 调整到 0。05。结果如图9所示。我们观察到运行时间有所增加。这可以解释如下：



(a) NBA

(b) CSRankings

图10：置信概率 δ 对运行时间的影响

具体如下。回顾 $\alpha = \widehat{p} + \eta$，其中 $\widehat{p}$ 是计算得到的 Sb 包含不稳定细化的估计概率，其定义见 命题 3.1。因此，当 α 固定时，随着 η 增加，$\widehat{p}$ 可接受的上界会降低，这意味着在 Sb 下采样得到的不稳定细化比例应当降低。要达到更低的$\widehat{p}$，可能需要增加样本数量，从而导致迭代次数增加，进而带来更高的计算时间。最后，图10 展示了 δ 取值从 0。01 到 0。1时的运行时间。如预期，随着 δ 值增加，即 置信水平较低时，运行时间较短。

6.4 稠密区域检测

接下来，我们从准确性和运行时间两个方面评估 Detect-Dense-Region。为此，我们使用 CSRankings 数据集和合成数据集，根据数据中密集区域的真实值评估输出质量。本实验不考虑 NBA 数据集，因为学习型排序函数会以截然不同的方式处理统计数据相似的项（例如，NBA 数据集中的乔尔·恩比德），导致排名中不存在明确的密集区域。

表3 列出了 CSRankings 中排名前十的大学及其计算得分。我们观察到，数据中不同群组之间的分数存在明显差异，表中对此进行了突出显示。具体而言，排名第1至第4的元组之间存在显著的分数差距，而排名第5至第8和第9至第10的元组之间，分数则更为接近，从而在数据中形成密集区域。例如，如果佐治亚理工学院在系统领域的调整后出版物数量仅少一项，其排名就会升至第8位。

Detect-Dense-Region 的评估成功识别出了密集区域。具体而言，前四个元组各自计算出的 k 值均为 0，表明它们相对于周围元组具有稳定性。对于斯坦福大学，Detect-Dense-Region 计算出的 k 值为 1，而人们或许会合理地预期$k = 2$，以便将华盛顿大学（UW）纳入其中。事实上，$k = 0$ 的局部稳定性值为 $7 \cdot 10^{-6}$，$k = 1$ 为 0.29，而$k = 2$ 为 0.83；即从 $k = 0$ 到 $k = 1$增加了 0.29，而从 $k = 1$ 到 $k = 2$ 增加了 0.54，增幅显著更大。

表3：根据 CSRanking 排出的前十名计算机科学系 [30]，（密集区域已突出显示）

	大学	AI	Sys	Thry	跨学科	得分 ↓
1st	CMU	71.4	11.9	21.1	13.8	19.53
2nd	UIUC	46.1	12.6	16.0	7.2	15.39
3rd	UCSD	31.6	9.0	10.1	10.3	13.00
4th	MIT	28.1	8.6	16.2	7.9	12.33
5th	佐治亚理工学院	28.5	7.8	6.9	10.2	11.58
6th	斯坦福大学	36.7	5.4	13.3	11.5	11.56
7th	密歇根大学	30.4	9.0	9.3	5.9	11.26
8th	UW	28.0	6.2	12.2	10.0	11.13
9th	UCB	23.2	7.4	15.9	6.4	10.69
10th	康奈尔大学	42.0	5.7	12.8	6.8	10.66

然而，由于 k 其他值之间的差异较小（均小于 0.15），聚类算法将前者归类为较大值。其余元组观察到的 k 值与表3所示的密集区域相符。这些结果凸显了 Detect-Dense-Region 在排名中检测密集区域的实用性，这是一种解决该问题的合理方法。考虑密集区域后，排名的局部稳定性便显而易见。具体而言，当选择 k 覆盖大学的密集区域时，我们发现除康奈尔大学外，所有大学的局部稳定性均至少为12。尤其是，康奈尔大学在其密集区域内的稳定性低于加州大学伯克利分校（UCB），因为 UCB 凭借其在系统出版物方面的相对优势而受益，康奈尔大学则没有，因此康奈尔大学排在第 11 位是合理的。

为了进一步评估 Detect-Dense-Region 的性能，我们使用了包含 $d = 2$ 和 100 个元组的合成数据集。该数据集通过生成密集区域而构建，因此为 k 值提供了真实值，这些值覆盖了元组所属的密集区域。我们对合成数据集中的全部 100 个元组运行了 Detect-Dense-Region，并将每个元组的输出 k 值与 k的真实值进行了比较。对于全部 100 个元组，Detect-Dense-Region 都能根据数据生成过程，正确推荐适合该密集区域的 k 值。

Detect-Dense-Region 的效率`Detect-Dense-Region`。为了展示 Detect-Dense-Region 的效率，我们将 Detect-Dense-Region 的运行时间与一个变体进行了比较，该变体使用 LStability 计算稳定性值。评估使用了合成数据集中的全部 100 个元组。Detect-Dense-Region 的总运行时间为 7 分钟，而使用 LStability 计算局部稳定性值时则需要 141 分钟。总体而言，我们观察到了20.3× 的提升；Detect-Dense-Region 的速度最高可提升 32×，平均提升 20.4×。

6.5 可扩展性与优化

接下来，我们将考察优化版本LSt（LSt）相较于第3节中介绍的基础版本（Basic）的可扩展性，并量化所提出的优化措施所带来的效果。

可扩展性。 我们使用合成数据集，在数据规模和属性数量不断增大的情况下评估了这些方法的可扩展性。图11 展示了 LStability 的优化版本和 Basic 版本的运行时间随数据规模变化的情况。正如预期，LSt 的表现优于 Basic。我们观察到，随着数据规模增大，Basic 的

运行时间呈线性增长，而数据规模对 LSt 运行时间的影响可以忽略不计。这证实，通过优化降低重新排序成本，可以消除元组独立排序函数对数据规模的依赖。

运行时间随属性数量变化的情况如图12所示。我们注意到，运行时间可能会受到元组稳定性的影响，即，稳定性低于阈值的元组τ_v通常由于提前终止而具有较短的运行时间。由于数据生成过程和元组选择具有随机性，每个经过评估的数据集很可能都包含稳定性高低不同的元组。因此，我们报告每个元组所观察到的最大运行时间。对于 Basic，运行时间随属性数量线性增加，因为每增加一个属性，例如，计算 $Min_{\prec}$ 一组细化操作的开销都会增加。此外，LSt 的运行时间随属性数量呈指数增长，因为必须执行更多迭代才能达到所需的 α。有趣的是，当属性数量超过 6 个时，LSt 的性能会下降，在最坏情况下变得比 Basic 更慢。在这些情况下，LSt 的执行次数接近分配的最大迭代次数，并且还会产生额外开销，例如，随着估计稳定区域缩小，执行拒绝采样的成本不断增加。不过，我们强调，这仅适用于最坏情况（图12展示了最大运行时间）。平均而言，LSt 11.8× 在属性数量为 6 至 10 个时更快。

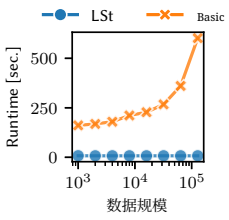


图11：数据规模

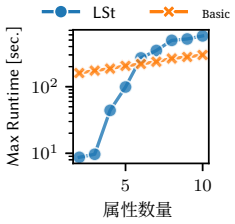
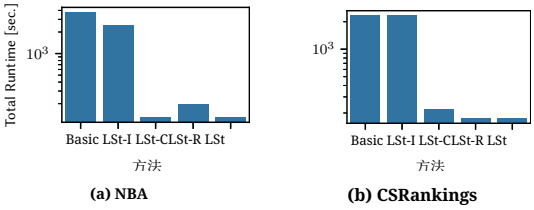


图12：属性数量



(a) NBA **(b) CSRankings**

图13：消融优化后各实验套件的总运行时间

消融研究。我们比较了不同变体的运行时间：LStability:（LSt）LStability 配备了所有优化；（LSt-I）表示LStability 禁用了有界 α 迭代优化的版本；（LSt-C）表示LStability 未启用缩减合理变更集合的优化；（LSt-R）表示LStability 未对重排序成本进行归约；（Basic）表示LStability 未启用任何优化，如第3节所述。为展示偏好带来的差异，我们报告了对所有前 10 元组计算稳定性值的总运行时间，其中 $0 \leq k \leq 10$ 取各算法变体对应的所有取值。结果见图13。

对于 NBA 数据集，Basic 慢 1.51× 。有趣的是，在 CSRankings 上，Basic 与 LSt-I 之间的性能差距可以忽略不计；而在 NBA 数据集上，Basic 慢 1.51× 。这是因为 Basic 和 LSt-I 抽取的样本数量相同，但 LSt-I 还采用了一种可降低重新排序成本的优化。由于前者使用的学习型排序函数成本更高，这种优化在 NBA 数据集上的效果比在 CSRankings 上更显著。研究表明，使用迭代计算来实现有界的 α 值是最有效的方法：在最佳情况下，单个元组的运行时间提升了 37× 倍；在 NBA 数据集上整体提升了 18.9× ，在 CSRankings 上则提升了 13.2× 。

合理变更优化的归约效果明显较小：NBA 数据集没有变化，而 CSRankings 的速度提升了 1.23× 。CSRankings 上效果更显著，是因为在该数据集中，对单个属性进行细微细化，就可能使元组在局部变得不稳定；而在 NBA 数据集中，学习排序模型会学习属性之间相对复杂的交互，因此通常需要细化多个属性，才能改变元组的排名。此外，我们注意到 CSRankings 中的排序函数具有单调性，因此可以更高效地计算归约后的合理变更。

最后，我们看到，在 NBA 数据集中，降低重排序成本这一优化带来了更大的收益， 1.51× 总运行时间有所下降；而在 CSRankings 中则没有变化。这是因为排序函数的运行时间取决于数据规模。对于 CSRankings，这项优化几乎没有影响，因为对于小型数据集，评估简单评分函数的运行时间与 LStability 的其他阶段相比微不足道。

6.6 局部与全局排序稳定性

接下来，我们将展示所提出的局部稳定性定义与 [3]中定义的（全局）排名稳定性之间的区别。我们针对二维情况实现了论文中描述的蒙特卡洛算法，并抽取 500,000 组函数权重，以确定在评分函数发生 a、b变更时，合成数据集的排名全局稳定性。该评分函数用于对 $\varphi(t) = a \cdot t.x + b \cdot t.y$进行排名，其中 $a,\ b \geq 0$。将 $a = 1,\ b = 1$设为相应值后得到的原始排名，其全局稳定性为 0.03（满分为 1）；而根据我们的定义，考虑到标记出的密集区域，大多数元组的局部稳定性相当高（平均为 0.45），这是由密集区域之间的间隔以及 第 6.1 节中描述的合理变化集合决定的。

相比之下，考虑一个包含 10 个元组的合成数据集变体，其中排名第 i的元组的每个属性值都比各属性中排名第 $i+1$ 的元组高 1。显然，这种排名被认为是完全稳定的，因为每个元组都支配所有排名在其后的元组。然而，根据我们的局部稳定性度量，对于未覆盖整个排名的 k 值，每个元组的稳定性至多为 0.08。

7 相关工作

排名稳定性。 排名稳定性已在一系列研究中得到探讨 [2, 3, 10, 16, 21, 46]。与我们工作最接近的是 [3], , 其

将稳定性视为以下属性：排名 相对于一族排名函数的稳定性，我们将其称为 全局 稳定性。从较高层面来看，稳定性度量输出对输入微小变化的响应程度。因此，任何具体的稳定性定义都取决于如何定义修改，以及输出效应的概念。对这些组成部分的具体选择决定了稳定性度量的语义。从技术上讲，[3] 将修改定义为对排名函数的更改，并将排名顺序中的任何变化 视为一种输出效应。相比之下，我们研究对数据 的修改，并将输出效应量化为给定元组的位置（即 其值 k）变化的幅度。概念上，[3] 中的稳定性概念旨在评估元组的（整体）排名出现的可能性，并可用于确定例如，排名函数是否经过精心挑选。相比之下，我们的定义旨在度量元组在原始排名函数下排名的接近程度；当没有可用得分，或得分与输入属性之间呈非线性关系时，这一点尤其重要。请注意，由于关注的是数据变化而非方法论，我们可以将排名过程视为黑箱，从而使我们的框架与模型无关。此外，我们的定义忽略密集区域内的变化，因此当排名中存在相似元组时，我们的度量会更加实用。

在 [10], 中研究了一种局部稳定性形式；然而，[10] 中的分析依赖于对排序函数和数据集的强假设，这些假设共同使得研究者能够以闭式形式刻画改变排名所需的扰动幅度。具体而言，除限制所考虑的排序函数类别外，他们的结果仅适用于完美赛季数据集——在这种数据集中，每支球队都恰好与其他每支球队交手一次，且没有爆冷：排名第一的球队击败所有其他球队，排名第二的球队只输给排名第一的球队，依此类推。这些假设使分析易于处理，却大幅限制了其普适性。将 [10]的框架扩展到更一般的数据集或其他排序函数并非易事，需要开展大量额外工作。相比之下，我们的方法与模型无关，且不对底层数据作任何假设。此外，[10]并未明确提供一种方法，用于衡量多个属性共同影响得分时元组的局部稳定性。最后，我们注意到，[10] 考虑的是恰好影响两个元组的扰动，而我们研究的是对单个元组所作的扰动。

排名解释。 除了排序稳定性之外，解释排名结果的问题也已得到广泛研究 [1, 11–13, 18, 30]。其中许多研究通过识别特征重要性来解释排名 [1, 11, 12, 18, 30]。尽管这些方法与稳定性有关，但它们并未像我们提出的方法那样，明确量化排名中各元组之间的间隔。在 [11, 12], 中，研究者将线性评分函数拟合到数据集，以匹配给定排名的前 k 项，或将单个元组排入前 k 名。由此得到的函数权重可解释为特征重要性的说明。ShaRP [30] 采用沙普利值，根据各个单独属性的贡献，解释为什么 某个项会获得相应的排名。同样，[1] 采用现有的特征重要性方法（例如，LIME [33]）来解释排名。[18] 的研究利用单调性假设推导特征重要性的重要性度量。另一方面，[13] 找到了足够的属性子集（称为溯因解释）；只要各方对这些属性达成一致，元组之间的排序就会相同。

机器学习中的鲁棒性验证。 验证机器学习模型的鲁棒性，近年来受到广泛关注 [4, 9, 24, 39, 45, 49, 50]。从本质上说，鲁棒性量化了输入受到扰动后，其预测发生改变的可能性 [9]。近期研究主要关注概率性的鲁棒性定义，与我们对 α-局部稳定性的定义类似；关于这一主题的最新综述，请参阅 [50] 。许多研究（例如，[4, 24, 39, 45, 49]）采用了与我们类似的框架，利用抽样和集中不等式来估计鲁棒性，并对其可靠性和准确性提供保证 [39, 45, 49]。不过，我们的定义与这类研究有一个关键区别：所考虑的扰动空间。通常，鲁棒性被视为一个 ℓ_p-球 [4] 上的函数 。这预设了一个针对扰动的成本函数，以便确定模型保持鲁棒时变化的最大成本上限。然而，我们对稳定区域边界的概念以一个更弱的假设取代了这一预设的成本函数：变化的成本是单调非递减的。这样便能纳入稳定区域中不属于任何 ℓ_p-球的部分（例如，“L”形稳定区域的臂部）。此外，这些研究中的推理任务彼此独立，不受其他得分的影响（也就是说，分类和回归）；而排名任务并非如此——局部稳定性不仅是排序函数的属性，也取决于排名中的其他元组（例如，在密集区域的情况下），因此需要针对这一任务进行特定的优化（例如，第4.2节）。

反事实解释。 反事实解释为如何针对不理想的分类结果实现补救提供了具体示例 [28, 31, 37, 38, 42, 44, 47]。这些方法大多着眼于通过尽量降低改变输入的成本来提供少数可操作的补救选项 [37, 44]。而我们则尝试刻画范围更广的补救选项，本质上是在不预先指定成本函数的情况下，找出最小补救。此外，许多方法（如 [28, 42] ）专为特定模型设计，并假设可以访问白箱，而我们基于抽样的方法则不依赖特定模型。

局部模型无关解释。 局部模型无关解释 [27, 30, 33, 34] 通过将任意给定模型视为黑箱，回答为何该模型会针对给定输入作出某一决策。在许多情况下，这类解释采用特征重要性度量的形式，例如 [27, 30, 33]。虽然特征重要性与单个属性的局部稳定性相关，但我们主要关注的是属性组合的变化。

8 结论

我们认识到，由于存在密集区域，其中少量不稳定性是可以合理预期并因此被容忍的，因此需要对排名中的稳定性采取一种局部视角。我们将这一概念形式化地定义为：其幅度不会引起大幅位置变化的细化，与借助领域专门知识选定的用户定义合理变更集合之间的比率。我们表明，在一般情况下精确计算这一变更集合是难解的，这促使我们提出局部稳定性的一种松弛定义。我们提出了 LStability,

一种基于采样的估计算法，并通过应用集中不等式给出了概率近似正确型保证。我们进一步提出了 Detect-Dense-Region，用于在排名中给定项的周围建议一个位置范围；由于该项与相似项相邻排名，我们可以合理预期微小变化会使该项发生移动。最后，我们通过对真实数据集的案例研究证明，我们对局部稳定性的定义能够提供有价值的洞见，证明了 Detect-Dense-Region 适用于其预期用途，并对影响我们的算法与优化性能的参数进行了实验分析。

我们的工作为进一步研究排名的局部稳定性开辟了诸多途径。未来工作可以考虑其他类别的细化，例如添加或删除数据，或要求细化满足约束（例如，拒绝约束或具有相关值的元组），以及处理分类数据。在某些情况下，将我们的框架扩展到分类属性或许是可行的，但这样做并非易事，而恰当地对细化幅度进行建模，取决于分类域是否具有自然顺序（例如，小、中、大尺寸）。

参考文献

[1] Hadis Anahideh与Nasrin Mohabbati-Kalejahi. 2022年. 全局排名的局部解释：竞争性排名的洞见. IEEE Access 10 (2022年), 30676–30693. doi:10.1109/ACCESS.2022.3159245[2] Abolfazl Asudeh与H. V. Jagadish. 2020年. 对数据集中项的公平评估与评分. VLDB基金会会刊 13, 12 (2020年), 3445–3448. doi:10.14778/3415478. 3415566[3] Abolfazl Asudeh, H. V. Jagadish, Gerome Miklau与Julia Stoyanovich. 2018. 论获得稳定排名. VLDB基金会会刊 12, 3 (2018), 237–250. doi:10. 14778/3291264.3291269[4] Teodora Baluta, Zheng Leong Chua, Kuldeep S. Meel与Prateek Saxena. 2021年. 面向深度神经网络的可扩展定量验证. 见第43届IEEE/ACM软件工程国际会议, ICSE 2021年, 西班牙马德里, 5月22-30日 *2021* 312 323 10 1109 43902 2021 00039. IEEE, –. doi: . /ICSE . . E 2025 [5] mery D. Berger. . CSRankings. https://csrankings.org/. [6] Emery D. Berger, Stephen M. Blackburn, Carla E. Brodley, H. V. Jagadish, Kathryn S. McKinley, Mario A. Nascimento, Minjeong Shin, Kuansan Wang与Lexing Xie. 2019年. GOTO排名被认为有帮助. ACM通讯 62, 7 (2019 29 30 10 1145 3332803), – . doi: . / Stephan Börzsönyi, Donald Kossmann与[7] Konrad Stocker. 2001. 天际线算子. 见第17届数据工程国际会议论文集, 2001年4月2-6日, 德国海德堡, Dimitrios Georgakopoulos与Alexander Buchmann (编). IEEE 计算机学会 , 421 430 10 1109 2001 914855 – . doi: . / 2010 [8] ICDE. . Karl Bringmann与Tobias Friedrich. . 高维几何对象并集与交集体积的近似. Comput. Geom. 43, 6-7 (2010 601 610 10 1016 2010 03 004), – . doi: . / [9] 2017 J.COMGEO. . . Nicholas Carlini与David A. Wagner. . 迈向神经网络稳健性评估. 见2017年IEEE安全与隐私研讨会, SP 2017年, 美国加利福尼亚州圣何塞, 2017年5月22-26日. IEEE计算机学会, 39–57. doi:10.1109/ SP.2017.49[10] Timothy P. Chartier, Erich Kreutzer, Amy N. Langville与Kathryn E. Pedings. 2011. 排名向量的敏感性与稳定性. SIAM科学计算期刊 33, 3 2011 1077 1102 10 1137 090772745 ( ), – . doi: . / Zixuan Chen, [11] Panagiotis Manolios与Mirek Riedewald. 2023年. 为何尚未：修正对个体不公平的Top-k排名. VLDB基金会会刊 16, 9 (2023 2377 2390 10 14778 3598581 3598606), – . doi: . / . [12] Zixuan Chen, Panagiotis Manolios与Mirek Riedewald. 2025年. 使用符号梯度下降合成排名的评分函数. 见第41届IEEE数据工程国际会议, ICDE 2025年, 23, 2025年 19 3849 3862 10 1109 65448 2025年 香港, 5月 - . IEEE, – . doi: . /ICDE 00287 . 2025 [13] Francesco Chiariello与João Marques-Silva. . 黑盒排名函数的形式化解释. 见人工智能中的逻辑 - 第19届欧洲会议, JELIA 2025年, 格鲁吉亚库塔伊西, 2025年9月1-4日, 论文集, 第一部分 (《计算机科学讲义》, 第16093卷), Giovanni Casini, Besik Dundua与Temur Kutsia (编). Springer, 28 44 10 1007 978 3 032 04587 4_3 – . doi: . / - - - - 2016 [14] Ben Cousins与Santosh S. Vempala. . 一种实用的体积算法. Math. Program. Comput. 8, 2 (2016年), 133–160. doi:10.1007/S12532-015-0097-Z [15] Mahsa Derakhshan, Negin Golrezaei, Vahideh Manshadi与Vahab Mirrokni. 2022年. 在线平台上的产品排名. Management Science 68, 6 (2022年). [16] Siddartha Devic、Aleksandra Korolova、David Kempe 和 Vatsal Sharan。2024。具有不确定预测的排名中的稳定性与多群体公平性。载于第41届机器学习国际会议（ICML 2024），奥地利维也纳，2024年7月21日至27日。OpenReview.net。https://openreview.net/forum?id=YiblhkVl2w [17] Walter D. Fisher。1958年。论最大同质性的分组。《美国统计学会会刊》53卷284期（1958年），789–798页。arXiv:https://www.tandfonline.com/doi/pdf/10.1080/01621459.1958.10501479 doi:10.1080/01621459.1958.10501479[18] Abraham Gale 和 Amélie Marian。2020年。解释单调排名函数。《VLDB基金会会刊》 14, 4 2020 640 652 10 14778 3436905 3436922 （），–。doi:。/。 2015 [19] Cunjing Ge 和 Feifei Ma。。一种估算凸多胞体体积的快速实用方法。载于算法前沿——第九届国际研讨会（FAW 2015），中国桂林，2015年7月3日至5日，论文集（《计算机科学讲义》，第9130卷），Jianxin Wang 和 Chee-Keng Yap（编）。Springer，52–65页。doi:10.1007/978-3-319-19647-3 6[20] Sahin Cem Geyik、Stuart Ambler 和 Krishnaram Kenthapadi。2019年。搜索与推荐系统中的公平感知排名及其在领英人才搜索中的应用。载于第25届ACM SIGKDD知识发现与数据挖掘国际会议论文集（KDD 2019），美国阿拉斯加州安克雷奇，2019年8月4日至8日，Ankur Teredesai、Vipin Kumar、Ying Li、Rómer Rosales、Evimaria Terzi 和 George Karypis（编）。ACM， 2221 2231 10 1145 3292500 3330691 –。doi:。/。[21] Yifan Guan、Abolfazl Asudeh、Pranav Mayuram、H. V. Jagadish、Julia Stoyanovich、Gerome Miklau 和 Gautam Das。2019年。MithraRanking：负责任的排序设计系统。载于2019年数据管理国际会议论文集（SIGMOD 2019大会），荷兰阿姆斯特丹，2019年6月30日至7月5日，Peter A. Boncz、Stefan Manegold、Anastasia Ailamaki、Amol Deshpande 和 Tim Kraska（编）。ACM， 1913 1916 10 1145 3299869 3320244 –。doi:。/。 2021 [22] Andreia P. Guerreiro、Carlos M. Fonseca 和 Luís Paquete。。超体积指标：计算问题与算法。《ACM计算调查》54卷6期，119 2021 42 10 1145 Article（7月），页数。doi:。/ 3453474 [23] Trevor Hastie、Robert Tibshirani、Jerome H Friedman 和 Jerome H Friedman。2009年。《统计学习的要素：数据挖掘、推断与预测》。第2卷。Springer。[24] Wei Huang、Xingyu Zhao、Gaojie Jin 和 Xiaowei Huang。2023年。SAFARI：用于可解释性稳健性的多功能高效评估。载于IEEE/CVF国际计算机视觉大会（ICCV 2023），法国巴黎，2023年10月1日至6日。IEEE，1988–1998页。doi:10.1109/ICCV51070.2023.00190[25] Jiarui Jin、Xianyu Chen、Weinan Zhang、Mengyue Yang、Yang Wang、Yali Du、Yong Yu 和 Jun Wang。2023年。以排列取代评分：一种用于学习排序的上下文集合到排列框架。载于第32届ACM国际信息与知识管理大会论文集（CIKM 2023），英国伯明翰，2023年10月21日至25日，Ingo Frommholz、Frank Hopfgartner、Mark Lee、Michael Oakes、Mounia Lalmas、Min Zhang 和 Rodrygo L. T. Santos（编）。ACM，1004–1013页。doi:10.1145/3583780.3615031[26] Guolin Ke、Qi Meng、Thomas Finley、Taifeng Wang、Wei Chen、Weidong Ma、Qiwei Ye 和 Tie-Yan Liu。2017年。LightGBM：一种高效的梯度提升决策树。载于《神经信息处理系统进展》第30卷：2017年神经信息处理系统年会，2017年12月4日至9日，美国加利福尼亚州长滩，Isabelle Guyon、Ulrike von Luxburg、Samy Bengio、Hanna M. Wallach、Rob Fergus、S. V. N. Vishwanathan 和 Roman Garnett（编）。3146–3154页。https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html[27] Scott M. Lundberg 和 Su-In Lee。2017年。一种解释模型预测的统一方法。载于《神经信息处理系统进展》第30卷：2017年神经信息处理系统年会，2017年12月4日至9日，美国加利福尼亚州长滩，Isabelle Guyon、Ulrike von Luxburg、Samy Bengio、Hanna M. Wallach、Rob Fergus、S. V. N. Vishwanathan 和 Roman Garnett（编）。4765–4774页。https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html[28] Ramaravind K Mothilal、Amit Sharma 和 Chenhao Tan。2020年。通过多样化的反事实解释来解释机器学习分类器。载于《2020年公平性、问责制与透明度大会论文集》。607–617页。[29] Jeremiah P Ostriker、Charlotte V Kuh 和 James A Voytuk。2011年。美国研究型博士项目的数据评估。美国国家科学院出版社。[30] Venetia Pliatsika、Joao Fonseca、Kateryna Akhynko、Ivan Shevchenko 和 Julia Stoyanovich。2024年。ShaRP：一种用于排名的新型特征重要性框架。arXiv:2401.16744[31] Rafael Poyiadzi、Kacper Sokol、Raúl Santos-Rodríguez、Tijl De Bie 和 Peter A. Flach。2020年。FACE：可行且可操作的反事实解释。载于AIES '20：AAAI/ACM人工智能、伦理与社会会议，美国纽约州纽约市，2020年2月7日至8日，Annette N. Markham、Julia Powles、Toby Walsh 和 Anne L. Washington（编）。ACM， 344 350 10 1145 3375627 3375850 –。doi:。/。Sports Reference。。—NBA球员统计：总计。检索于2024年7月29日，网址：http://basketball-reference.com/leagues/NBA_2024_totals.html [33] Marco Túlio Ribeiro、Sameer Singh 和 Carlos Guestrin。2016年。"我为什么应该相信你？"：解释任意分类器的预测。载于 *2023 24* [32]

第22届ACM SIGKDD知识发现与数据挖掘国际会议论文集，美国加利福尼亚州旧金山，2016年8月13日至17日，Balaji Krishnapuram、Mohak Shah、Alexander J. Smola、Charu C. Aggarwal、Dou Shen 和 Rajeev Rastogi（编）。ACM，1135 1144 10 1145 2939672 2939778 –。doi:。/。Marco Túlio [34] Ribeiro、Sameer Singh 和 Carlos Guestrin。2018。Anchors：高精度模型无关解释。载于第三十二届AAAI人工智能会议论文集（AAAI-18）、第30届人工智能创新应用大会（IAAI-18）以及第八届AAAI人工智能教育进展研讨会（EAAI-18），美国路易斯安那州新奥尔良，2018年2月2日至7日，Sheila A. McIlraith 和 Kilian Q. Weinberger（编）。AAAI出版社，1527–1535页。doi:10.1609/AAAI.V32I1.11491[35] The Ringer。2024年。NBA排名。检索于2024年7月29日，网址：https:// nbarankings.theringer.com [36] Christian P. Robert 和 George Casella。2004。蒙特卡洛统计方法。Springer。doi: 10 1007 978 1 4757 4145 2 。/ - - - - Maximilian Schleich、[37] Zixuan Geng、Yihong Zhang 和 Dan Suciu。2021年。GeCo：实时高质量反事实解释。《VLDB基金会会刊》14卷9期（2021年），1681–1693页。doi:10.14778/3461535.3461555[38] Shubham Sharma、Jette Henderson 和 Joydeep Ghosh。2020年。CERTIFAI：为黑箱模型提供解释并分析公平性与稳健性的通用框架。载于AIES '20：AAAI/ACM人工智能、伦理与社会会议，美国纽约州纽约市，2020年2月7日至8日，Annette N. Markham、Julia Powles、Toby Walsh 和 Anne L. Washington（编）。ACM，166–172页。doi:10.1145/ 3375627.3375812[39] Karim Tit、Teddy Furon 和 Mathias Rousset。2021年。神经网络抗损坏鲁棒性的高效统计评估。载于《神经信息处理系统进展》第34卷：2021年神经信息处理系统年会（NeurIPS 2021），2021年12月6日至14日，线上举行，Marc'Aurelio Ranzato、Alina Beygelzimer、Yann N. Dauphin、Percy Liang 和 Jennifer Wortman Vaughan（编）。9253–9263页。https://proceedings.neurips.cc/paper/2021/hash/4d215ab7508a3e089af43fb605dd27d1-Abstract.html[40] Raluca M. Ursu。2018。排名的力量：量化排名对在线消费者搜索与购买决策的影响。《营销科学》37卷4期（ 2018 530 552 48748468 ），第–页。[41] 1979 https://www.jstor.org/stable/ Leslie G. Valiant。。枚举与可靠性问题的复杂性。《SIAM计算杂志》8卷3期（1979年），410–421页。doi:10.1137/0208032

[42] Peter M. VanNostrand、Huayi Zhang、Dennis M. Hofmann 和 Elke A. Rundensteiner。2023。FACET：稳健反事实解释分析。《ACM 管理数据学报》1, 4（2023），242:1–242:27。doi:10.1145/3626729[43] Sahil Verma、Varich Boonsanong、Minh Hoang、Keegan Hines、John Dickerson 和 Chirag Shah。2024。机器学习的反事实解释与算法追索：综述。《ACM 计算机调查》56, 12，文章 312（2024年10月），页码。doi: 。 / Sandra Wachter 、 Brent D. Mittelstadt 和 Chris Russell。2017。无需打开黑箱的反事实解释：自动化决策与通用数据保护条例。CoRR abs/1711.00399（2017）。arXiv:1711.00399 http://arxiv.org/abs/1711. 00399[45] Lily Weng、Pin-Yu Chen、Lam M. Nguyen、Mark S. Squillante、Akhilan Boopa- thy、Ivan V. Oseledets 和 Luca Daniel。2019。PROVEN：采用概率方法验证神经网络的稳健性。载于《第36届国际机器学习会议论文集》，ICML 2019，2019年6月9日至15日，美国加利福尼亚州长滩（《机器学习研究论文集》，第97卷），Kamalika Chaudhuri 和 Ruslan Salakhutdinov（编）。PMLR，6727–6736。http://proceedings.mlr.press/v97/weng19a.html [46] Ke Yang、Julia Stoyanovich、Abolfazl Asudeh、Bill Howe、H. V. Jagadish 和 Gerome Miklau。2018。排名的营养标签。载于《2018年数据管理国际会议论文集》，SIGMOD 2018会议，美国得克萨斯州休斯敦，2018年6月10日至15日，Gautam Das、Christopher M. Jermaine 和 Philip A. Bernstein（编）。ACM，1773 1776 10 1145 3183713 3193568 –。doi: 。 / 。[47] 2022 Wenzhuo Yang、Jia Li、Caiming Xiong 和 Steven C. H. Hoi。。MACE：用于反事实解释的高效模型无关框架。CoRR abs/2205 15540 2022 2205 15540 10 48550 2205 15540 。 （）。arXiv: 。[48] 2023 doi: 。/ARXIV. 。Meike Zehlike、Ke Yang 和 Julia Stoyanovich。。排序公平性，第一部分：基于分数的排序。《ACM 计算机调查》55, 6（2023），118:1–118:36。doi:10.1145/3533379[49] Tianle Zhang、Wenjie Ruan 和 Jonathan E. Fieldsend。2022。PRoA：针对函数扰动的概率稳健性评估。Springer-Verlag，柏林，海德堡，154 170 10 1007 978 3 031 26409 2_10 –。doi: 。/ - - - - Xingyu [50] 2025 Zhao。。深度学习中的概率稳健性：简明而全面的指南。arXiv:2502.14833 [cs.LG] https://arxiv.org/abs/2502.14833

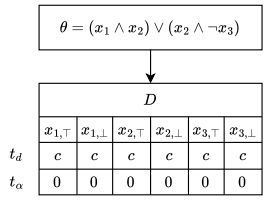


图14：构造的数据库示例 D ($c > 1$) 由析取范式公式构造 θ，用于引理证明中的归约：引理 A.1

计算 k-$\mathrm{SB}^t_{f(D)}$ 的困难性

我们首先证明

引理 A.1。 给定一个数据库 D、一个元组 $t \in D$、一个排序函数 f 和一个值 k，计算 $|k\text{-}SB^t_{f(D)}|$ 是 $\#P$ 困难。

证明。我们通过从 #DNF 进行保数归约来证明这一点。简而言之，#DNF 计算析取范式布尔公式的满足赋值数量。此外，#DNF 是一个众所周知的 #P 完全问题 [41]。设 θ为一个析取范式公式，由涉及一组变量 $x_1, \ldots, x_n$ 的子句构成。令 α 表示这些变量的一个赋值，其中，如果α是满足公式 θ 的变量赋值，则称 $\alpha \models \theta$，否则称$\alpha \not\models \theta$。接着，我们令 $\alpha(x_i)$ 表示变量 x_i 的赋值。

现在，我们将为问题构造归约。令 D 为一个由两个 $2n$维元组组成的数据库： v是一个将映射到 θ中变量赋值 α的元组；而 w是一个“虚拟”元组，当 v 是满足赋值时，它将与 v 交换排名。我们将 v 初始化为 $2n$维零向量，并将 w 初始化为 $2n$维向量，其每个分量都是常数 $c > 1$。直观地说， $i \in [n]$ 中的每个原始变量 x_i 在 v中由两个属性表示；当赋给 x_i 的值分别为 $\top$ 或 $\bot$时，对应的属性设为 1。正如我们很快将看到的，这种分离是必要的，否则，最小 k不稳定细化集合中的细化所要求的包含条件将变得棘手。

现在，我们定义一个映射 φ，它将元组 $t \in D$ 映射为赋值 α，或者在无法映射为有效赋值时映射为特殊值 $\times$。对于 θ 中的变量 x_i和元组 $t \in D$， 令y_i、$\top$和y_i、$\bot$分别表示 t中对应于对x_i赋真值和假值的属性。我们首先处理映射无效的情况。如果对于任意 x_i，对应的y_i、$\top$、 y_i、$\bot$值不属于$\{0, 1\}$，那么 $\varphi(t) = \times$。此外，如果任意y_i、$\top$、y_i、$\bot$均为 0 或均为 1，那么 $\varphi(t) = \times$。现在，映射中只剩下处理有效赋值的情况。若上述情况均未发生，则 $\varphi(t)$是赋值 α，并且对于 θ中的每个变量 x_i，若y_i，则 $\alpha(x_i) = \top$；否则， $\alpha(x_i) = \bot$。

现在，我们可以构造归约的核心：排序函数。令 $f(D)$ 返回如下排名： $[-1, 1]$ 中所有属性值均为 $t \in D$、且满足 $\varphi(t) \models \theta$ 的元组，排在所有属性值均设为 c 的元组之前；而这些元组又排在 $[-1, 1]$ 中所有属性值均为 φ、且满足 $t(\not\models \theta)$ 的元组之前。因此很容易看出，如果我们将 v 细化为 $v_\prime$，那么当且仅当 $v_\prime$ 是满足赋值时， $v_\prime$ 才会在 $f(D_{v \to v'})$ 中排在 w 之前。现在令 $k = 0$，我们有任意

在 k-$\mathrm{SB}^v_{f(D)}$ 中的细化都可以映射为 θ 的一个满足赋值。

剩下要证明的只有 k-$\mathrm{SB}^v_{f(D)}$ 包含 θ 的所有满足赋值。回顾一下，当且仅当存在一个细化元组 $v' \in \varepsilon(v)$，使得 $\varphi(v') \models \theta$ 时， $\varepsilon$of v 的细化是不稳定的。因此，我们只需证明不存在包含于其他细化中的 0-不稳定细化；根据定义，这样一来，每个 0- 不稳定细化都会包含于 k-$\mathrm{SB}^t_f(D)$。为此，设 ε 是 v 在 $f(D)$ 上的一个 0- 不稳定细化，并反设存在一个不同的细化 ε'，它是 v 在 $f(D)$ 上的 0-不稳定细化，且 $\varepsilon' \prec \varepsilon$。由于 $\varepsilon' \prec \varepsilon$， ε' 的属性上各细化的幅度中，必有一个小于 ε 中对应的幅度。由于 ε 的所有分量都为 0 或 1（这是因为它是一个细化，且存在一个细化元组可映射到有效赋值），并且细化幅度的下界为 0，因此差异必然是： ε 的某个属性上的幅度为 1，而 ε' 中对应的幅度为 0。不失一般性，假设这是对应于某个 $y_{i,\top}$ 的幅度。注意，由于在 ε 中，$y_{i,\top}$ 的幅度为 1，因此 $y_{i,\bot}$ 必须为 0，才能包含一个映射到有效赋值的细化元组。然而，由于 $\varepsilon' \prec \varepsilon$， ε' 中 $y_{i,\bot}$ 的幅度必须为 0，而 $y_{i,\top}$ 也必须为 0。这与假设 ε' 是一个 0-不稳定细化相矛盾，因为它的细化集包含一个映射到满足赋值（或任何有效赋值）的元组。因此，不可能存在一个细化 ε' 包含于 ε 且同时为 0-稳定；也就是说， v 在 $f(D)$ 上的每个 0-稳定细化都属于 k-$\mathrm{SB}^t_f(D)$。

由此，我们可以得出：每个赋值 α 使得$\alpha \models \theta$都与 k-$\mathrm{SB}^v_f(D)$ 中的一个 0-不稳定细化一一对应。因此，给定一个用于计算 $|k$-$\mathrm{SB}^t_f(D)|$的预言机，我们便能够在多项式时间内回答 #DNF。

我们注意到，对于 k的任何取值，这都是困难的，因为我们只需根据需要添加任意多个虚拟元组即可。□

现在我们可以进行证明

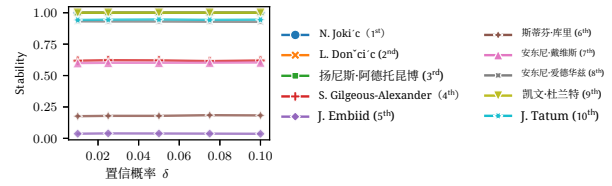
定理 A.2（定理2.14）。除非 $FP = \#P$，否则不存在用于计算 k-$SB^t_f(D)$ 的多项式时间算法，其中给定数据库 D、元组 $t \in D$、排序函数 f和一个值 k。

证明。我们通过证明逆否命题来证明这一点：如果存在一个用于计算 k-$\mathrm{SB}^t_f(D)$ 的多项式时间算法，那么 $\mathrm{FP} = \#\mathrm{P}$。

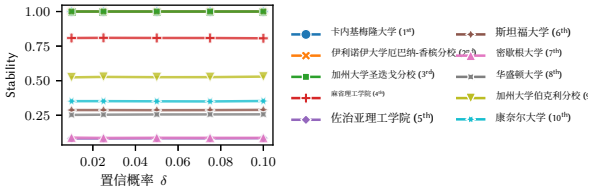
一个用于计算 k-$\mathrm{SB}^t_f(D)$ 的多项式时间算法意味着，计算 $|k$-$\mathrm{SB}^t_f(D)|$ 同样可以在多项式时间内完成，因为我们只需运行该算法，然后计算输出 k-$\mathrm{SB}^t_f(D)$ 中的细化数量（根据我们对运行时间的假设，其大小有多项式上界）。然而，根据引理 A.1，所有 #P 问题都可以通过 #DNF 在多项式时间内归约为计算 $|k$-$\mathrm{SB}^t_f(D)|$。因此，这样的算法意味着 $\mathrm{FP} = \#\mathrm{P}$。□

B 二分搜索：缩小合理变化集合

排序函数的一个典型特征是单调性，即，元组的某个属性得到改善后，其排名应当只能

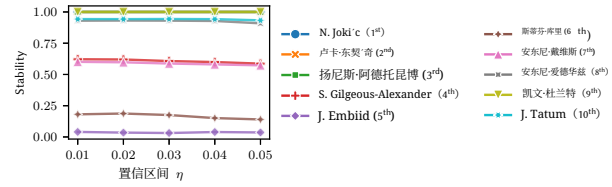


(a) NBA

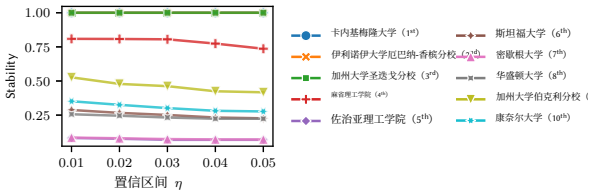


(b) CSRankings

图16：置信概率 δ 对估计稳定性的影响

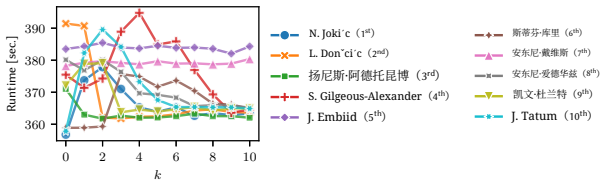


(a) NBA

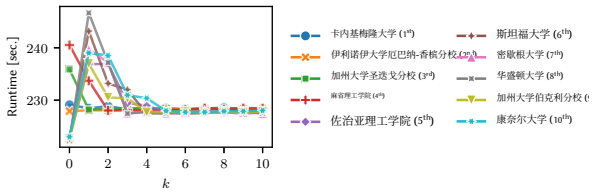


(b) CSRankings

图17：置信区间 η 对估计稳定性的影响



(a) NBA



(b) CSRankings

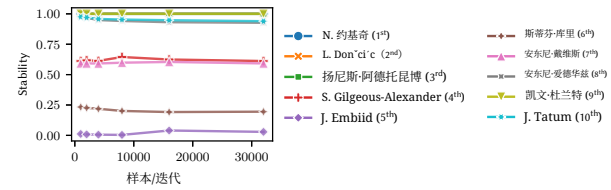
图18：不同 k 下 Basic LStability 的运行时间

得到提升。我们通常可以作出这样的假设，因为单调排序函数很常用，它们体现了这样一种预期：属性是效用的代理，因此通过改善（提高）这些属性，元组的排名应当只能提升 [18]。

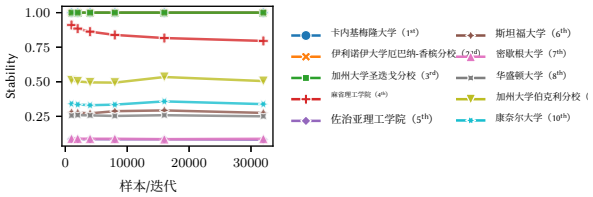
回顾一下，为了缩减合理变更，我们会使用一部分采样预算，对每个属性 a_i的单维细化 ε_i进行抽样，并令 ε_i^*表示样本中满足以下条件的 ε_i 的最小（绝对）值： $\varepsilon(t)$对于 $f(D)$是 k-不稳定的。如果给定的排序函数是单调的，我们就能在对数时间内，为属性 a_i 上的 ε_i^*找到一个紧确的下界（相差一个常数因子）。形式化地说，当且仅当对于任意 t、 t'，若存在某个属性 a，使得 t'在该属性上优于 t，即 $t'.a \geq t.a$，且在所有其他属性上相等，则对于每个包含 t、 t'的数据库 D，都有$f(D)[t'] \leq f(D)[t]$，此时排序函数 f是单调的（回顾一下，我们假设排名越低越好） [18]。根据这一定义，对于单调排序函数 f和细化ε，判断每个细化 $\varepsilon' \leq \varepsilon$ 对于$f(D)$上的 t 是否为 k-稳定，就变得简单了。

为此，对于细化 ε，令 ε^+ 和 ε^- 分别表示与 ε幅度相同、但所有分量均为正或均为负的细化。如果 $\Delta_{f(D)}(t, \varepsilon^+(t)) \leq k$以及$\Delta_{f(D)}(t, \varepsilon^-(t)) \leq k$，那么由于 f的单调性，每个 $\varepsilon' \leq \varepsilon$ 对于 t 在 $f(D)$ 上都是 k-稳定的。要找出属性 a_i 上 ε_i^*的紧下界，只需对属性 a_i 上细化的可能幅度进行二分搜索（这些幅度通过对 0 与 RC_i之间的范围进行离散化得到），从而找到仅通过细化 a_i 所产生的 k-稳定细化与 k-不稳定细化之间的边界。

C 补充实验



(a) NBA



(b) CSRankings

图15：构造步骤每轮抽取的样本数对估计稳定性的影响

图15 至 17 表明，LStability 的构造和验证步骤中的参数不会大幅改变其对局部稳定性的估计。在某些情况下，稳定性可能会略有下降，例如，麻省理工学院（MIT）在图8 中，或加利福尼亚大学伯克利分校（UCB）在图17中。当获取了额外样本时（无论是有意为之，还是因为尚未在 α 达到所需阈值），就会出现这种情况，因此对稳定区域边界的估计会得到进一步细化。

图18 显示，基本方法对 k 的依赖与优化版本类似。具体而言，k值越大，运行时间越短，因为估计稳定区域较大时，验证步骤耗时更少（这也使拒绝采样耗时更少）。

收到日期：2007年2月20日；修订日期：2009年3月12日；接受日期：2009年6月5日