基于 Opendigger 指标数据的项目健康度分析

1. **项目背景**

在现代软件开发进程中，开源软件占据着举足轻重的地位。它不仅助力技术的广泛共享与创新，还能显著削减开发成本，提升软件质量。GitHub 作为全球首屈一指的代码托管平台，汇聚了海量开源项目，其涵盖的编程语言与应用领域极为广泛。其中，Top 300 项目凭借较高的星标数、活跃的社区以及广泛的使用基础，成为开发者与企业瞩目的焦点。 然而，GitHub 原生提供的诸如星标数、Fork 数等指标，在反映项目真实健康状况方面存在明显局限。开发者若单纯依赖这些表面指标，极有可能错失潜力巨大但知名度欠佳的项目，或者误选社区活跃度低迷、维护不善的项目，最终对使用和贡献效率造成负面影响。

有鉴于此，本项目引入 Opendigger 指标数据，融合多维度评估视角，涵盖项目活跃度、问题解决能力、开发效率、社区关注度等方面，力求为 GitHub 上的热门项目（Top 300）打造一套更为全面、科学的健康度衡量方案。

1. **设计目标**
2. 多维度健康度分析：

充分利用 Opendigger 提供的项目健康度指标，包括活跃人数，PR生命周期，贡献者数量等等，全方位综合评估项目的实际健康状况。

1. 直观的可视化展示：为用户精心打造清晰直观的可视化大屏，

涵盖项目排名、各维度评分、趋势图、对比分析等内容，助力开发者及企业快速做出决策或选型。

1. **数据分析方法**
2. 数据读取：

本项目选用 top300 数据文件夹，其文件结构呈现大文件夹套小文件夹，小文件夹内嵌套多个项目小文件夹的形式，且每个项目文件夹包含多个 json 文件。为此，先定义递归函数查找对应 json 文件，再遍历项目文件，以防在小文件夹内读取一个项目数据后便停止读取，确保数据的完整性。

1. 数据分析
2. **社区活跃度**

社区活跃度由三个子指标综合而成，各指标详情如下：

1. 活跃度数值：

数据来源：activity.json

计算方法：针对每个项目，先计算其每年 activity 数值的平均值，该值可理解为项目在当年的平均活跃度。于同一年度范围内，对所有项目的活跃度平均值实施最大 - 最小归一化（Min - Max Normalization），使其映射至统一区间，进而得到“活跃度数值”得分。

含义：此得分直观反映项目一年内的综合活跃度高低，活跃度越高，意味着社区互动、贡献越频繁。

1. 粘度：

数据来源：active\_dates\_and\_times.json 和 activity.json（用于获取日活跃用户数 DAU、月活跃用户数 MAU）

计算方法：首先计算日活跃用户数（DAU）与月活跃用户数（MAU），得出年度内每个时间点的 DAU/MAU 比率。 - 进一步统计这些比率在当年内的方差，方差越小，表明粘度在一年内较为平稳，项目用户活跃度能够持续维持。 - 最后将粘度及其方差的结果转化为标准化分数。

含义：DAU/MAU 比率越高，粘度越强，说明社区用户天天或经常参与项目；年度方差越小，意味着这种粘性能够长期稳定保持，而非大起大落。

1. 参与者数量：

数据来源：participants.json

计算方法：统计每个项目在一年内实际参与项目的用户总数，并对该数值进行归一化处理。

含义：该指标反映在特定年份里，有多少不同的开发者或用户投身于项目之中，数量越多、分布越稳定，项目的社区基础就越坚实。

1. 综合得分：将上述三个子指标（活跃度数值、粘度、参与者数量）各自归一化后，赋予相同权重，合并得到社区活跃度分数。
2. **开发效率**
3. **指标一：PR 生命周期**

数据来源：change\_request\_age.json：表示合并时长（从创建到合并的时间），change\_request\_resolution\_duration.json：表示关闭时长（从创建到关闭的时间）。

计算思路：

分别处理合并与关闭时长：从 change\_request\_age.json 获取合并时长数组；从change\_request\_resolution\_duration.json 获取关闭时长数组。

逐年汇总： 将该年度的合并时长/关闭时长进行求和，再平均，得到单个项目在这一时间范围内的总平均合并时长、总平均关闭时长。 合并为“生命周期”一项，使用加权合成，将合并时长、关闭时长合成为一个综合“PR 生命周期”指标。生命周期= 0.5×(合并时长) + 0.5×(关闭时长) ，这里各占 50%,时间越短代表项目在处理 PR 时更高效,开发效率越高。

归一化处理： 由于时长越短越好，采用反向的最大 - 最小归一化，先进行正常的 Min - Max 再用 1 - x 做反转，将结果映射到 0～100 分区间，时间越长分数越低。

1. **指标二：PR 响应时长：**

数据来源：change request response time.json

计算思路：获取每个 PR 的响应时间（按照年/月分组存储）。将同一年的响应时长累计，得到该项目在该年度/月份的总响应时长。同样，时间越短表示响应效率越高，因此要做反向的归一化处理，最终映射到 0～100 分区间。

1. **综合计算：项目开发效率得分：**

有了“PR 生命周期”和“PR 响应时长”两大指标后，按照设定的权重进行合成： - PR 响应时长占 50%； - PR 生命周期中又包含两个子指标（合并时长、关闭时长），各占 25%（或先行加权成一个指标，再占剩余 50%）。

1. **社区关注度**

数据来源：来源于“technical\_fork.json”“stars.json”以及“attention.json”

计算思路：按年汇总数据：针对从文件中读取到的如“fork\_data”和“stars\_data”这类月度数据，按照年份进行汇总。具体做法是从日期字符串中提取出年份，若该年份首次出现，则初始化对应年份的数据值为 0，之后将属于该年份的各月度数据进行累加，从而得到每年的数据情况。

计算社区关注度（attention）：先获取“fork\_data”和“stars\_data”共同涉及的年份集合，并进行排序。对于每一年，获取该年对应的“fork\_value”和“stars\_value”，在计算增长率时，若为第一年则增长率设为 0，非第一年则根据当年与上一年“fork\_value”和“stars\_value”的总和差值来计算增长率。最后，按照设定的权重（“fork”占 0.4、“stars”占 0.4、“growth rate”占 0.2）来计算该年的“attention”数值，以此体现社区关注度情况。

1. **问题解决能力**

**1.指标一：关闭率：**

数据来源：Issue 数量与关闭率：issues\_new.json：记录每月/每季度新增 Issue 的数量；issues\_closed.json：记录对应时间段内已关闭的 Issue 数量；

计算思路：关闭率（Close Rate）：Close Rate = 已关闭的 Issue 数 / 新增的 Issue 数。

统计某时间段（如月度或季度）内新增加的 Issue 及其关闭数量。

意义：高关闭率通常表明项目能够及时、有效地解决问题；低关闭率可能代表项目积压了大量未处理的 Issue。

**2.指标二 平均响应时长（Average Response Time）：**

数据来源：Issue 响应时长： issue\_response\_time.json：每个 Issue 的首次响应所经历的时间。

计算思路：从 issue\_response\_time.json 中提取 Issue 的首次响应时间。 对该时间值取平均，得出“平均响应时长”。

意义：若响应时间过长，表明项目维护团队可能人手不足或资源调配不合理；响应时间越短则意味着对社区问题反馈的处理更及时。

指标三 平均生命周期（Average Lifecycle）

数据来源：Issue 生命周期：issue\_age.json、issue\_resolution\_duration.json：分别可以表示从报告到解决的时长，或从报告到关闭/合并的总时间。

计算方法：使用 issue\_age.json 或 issue\_resolution\_duration.json，统计 Issue 从报告到解决/关闭的时间。同样可对该时长取平均值。

意义：生命周期越短，说明项目团队可以更快地修复 Bug 或实现功能需求，高效的开发维护能力能增强用户对项目的信赖。 综合计算：

问题解决能力分数：

加权组合：设定权重 w1, w2, w3 分别对应关闭率、平均响应时长、平均生命周期。由于响应时长和生命周期是“越短越好”，而关闭率是“越高越好”，需要事先对其做反向或正向归一化处理。

含义：结果分数越高，表示该项目在处理社区问题时效率更高、响应更及时、解决率更佳；分数偏低则需要关注团队协作、资源分配或项目规划等方面是否存在改进空间。

1. **健康分数**
2. 指标标准化（Min - Max Scaling）：

在获取完整且无重复项目名称的数据后，每个项目拥有四个原始指标分值。鉴于这些分值可能存在数量级或量纲差异，为确保计算可比性，需先进行“标准化”或“归一化”。Min - Max Scaling 计算简便、范围直观统一（0 - 100），便于后续计算与比较。

1. 处理零值：在几何平均计算中，若任意一个指标分数为零，则整个健康分数会变为零。但实际业务场景下，分数为零可能源于数据缺失或统计口径问题，并非项目在此指标上毫无作为。为此，代码将所有 0 替换为一个极小正数（如 1e - 6），使几何平均在该指标上的贡献极低，但避免整体分数直接归零。
2. 几何平均（Geometric Mean）计算健康分数：四个指标经标准化和零值处理后，记为 A, B, C, D，采用几何平均计算健康分数（Health\_Score\_Geometric）：Health\_Score\_Geometric = ⁴√(A × B × C × D)。几何平均特点为：若任意一个指标分数较低，整体结果会被大幅拉低，比算术平均更“敏感”；只有四个指标都较高时，最终分数才会取得较高值。
3. 将健康分数线性缩放到区间 30 - 90：即便使用几何平均，不同项目的健康分数仍可能集中于较小范围。为便于人眼区分与后续可视化，在几何平均结果基础上，做一次线性缩放，将分数分布在 30 - 90 的区间内。缩放公式如下：Health\_Score\_Scaled = 30 + 60 × (Health\_Score\_Geometric - min(GM)) / (max(GM) - min(GM))，其中 min(GM) 与 max(GM) 分别为几何平均分数的最小值和最大值，Health\_Score\_Scaled 因而分布于 30 到 90 之间，增强了分数的区分度和可读性。
4. **异常监测**
5. **整体计算思路：**

数据来源：每个项目在不同年份有两项指标，即参与度分数（attention\_score）和活跃度分数（activity\_score）。以“项目名称”与“年份”为依据，合并获取同一项目、同一年份的这两项数值。

连续三年检测：判断某项目在连续三年的时间范围内，每一年是否都在“显著下降”（设定为≥10%）。这里的“三年”实际形成两个年度区间：第 1 年 → 第 2 年和第 2 年 → 第 3 年。

具体标准：对于该项目，在这两个相邻年度区间中，“参与度分数”与“活跃度分数”都要有超过 10% 的降幅，才视为“三年连续下降”。降幅计算：降幅 = (新值 - 旧值) / 旧值，若该比值 ≤ - 0.10（- 10%），则认为该年度有显著下降。

异常判定：若一个项目在任意一个“三年窗口”上，参与度与活跃度都满足连续两次（即两年间隔）的≥10%降幅，则认定该项目出现了异常。若该项目完全没有这样的“三连降”时段，则正常，无需预警。

危机指数：在检测到符合条件的“三年窗口”后，可额外定义一个“危机指数”来衡量严重程度。使用第 1 年的数值减去第 3 年的数值（若结果为负则置 0），并分别对“参与度”与“活跃度”累加，得到一个简单的“总降幅”。数值越大，说明从第一年至第三年的下跌越明显，危机越大。

最后我们设置一个异常指数的阈值，展示出异常指数超过阈值的项目。

1. **可视化大屏设计**

可视化大屏

项目活跃度的可视化：

通过柱状图展示各项目的平均活跃系数，并按照字母顺序呈现。

数据洞察：绝大部分项目的活跃系数集中在30-50之间



通过折线图展示平均活跃系数排名前十的项目的年度活跃系数变化趋势。

数据洞察：从2016年至2023年，大多数技术或服务的活跃度呈现出波动上升的趋势。某些技术或服务在某些年份的使用量出现下降，反映了市场竞争和技术迭代的复杂性。此外，不难发现2019年是多个技术或服务使用量达到或接近峰值的一年，反映了该年份在技术市场上的活跃度和竞争力。





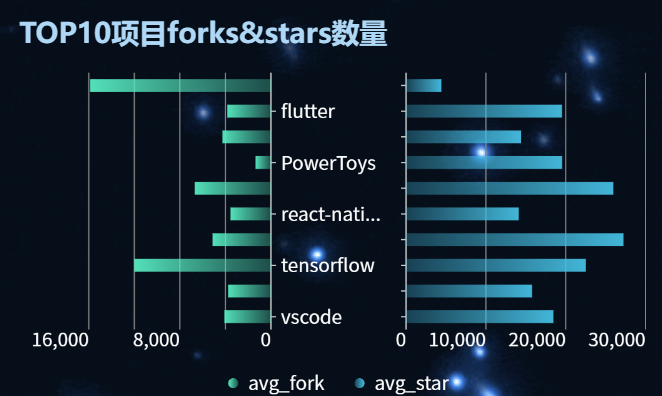
通过柱状图展现从2015年至2023年期间年平均参与者数量最多的TOP10项目

数据洞察：在年平均参与者数量最多的TOP10项目，绝大多数项目的参与者数量都是分布在1000上下，只有极个别项目参与者数量会到达1500以上。



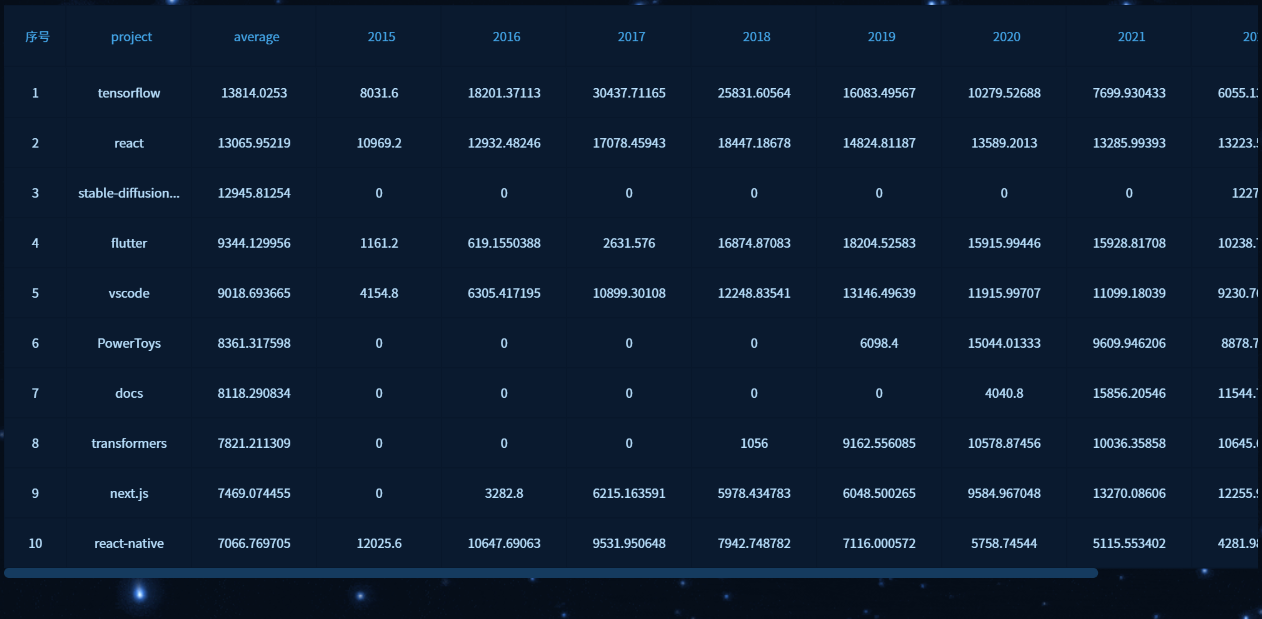
社区关注度可视化：

通过对称条形图分别展示项目的forks数量和stars数量。该图表能够清晰展示哪些项目在社区中获得了最多的关注和支持，并分析forks和stars数量之间的相关性，从而揭示社区对项目的关注度和认可度。



通过表格展现出年均社区关注度的TOP10项目的年度关注度变化。

数据洞察：stable - diffusion 的数据波动最为明显，从 2015 年的 12945.81254 下降到 2021 年的 1277.0，可能反映了该项目在这几年间经历了重大变化或发展方向的调整。Tensorflow 的数据在各年份间波动相对较小，在一定范围内变化，反映出该项目可能具有较为稳定的发展态势。其中，不乏有许多新起之秀，尽管起始时间较晚，其受社区关注度也十分的高。

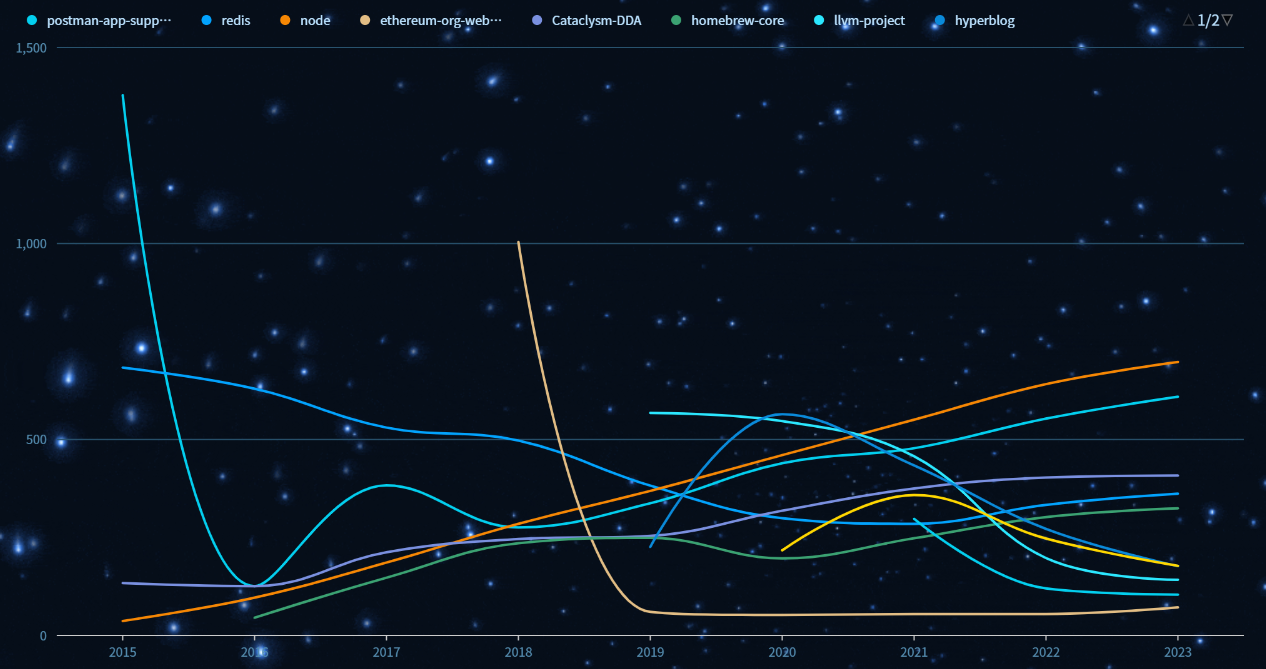


开发效率可视化：

通过柱状图展示年均开发效率的TOP10项目。



通过折线图展示出年均开发效率TOP10项目的年度开发效率变化趋势。折线图展现了这些高效项目在不同年度的开发效率变化，帮助我们理解哪些项目在某些年份表现更突出，哪些则有所下降，进而反映出项目管理、团队协作及技术进步等因素的影响，同时为项目团队提供参考。

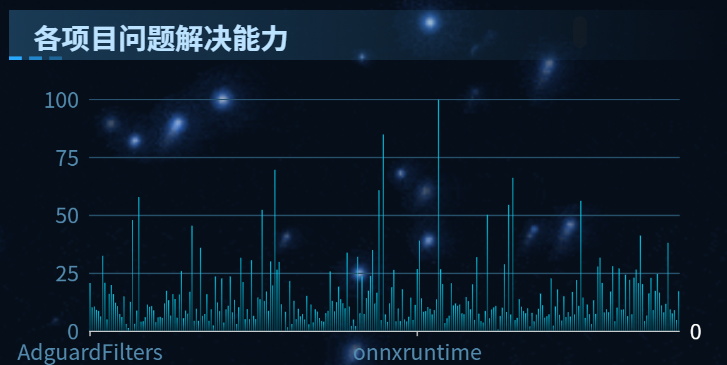


问题解决能力可视化：

通过柱状图展示出问题解决能力TOP10的项目。此图能够直观展示哪些项目在处理问题和解决用户反馈方面最为高效，从而为项目优化和用户体验改进提供参考。



通过柱状图展示各项目的问题解决能力。通过该柱状图，可以比较不同项目在问题解决上的能力差异，帮助识别哪些项目需要在问题反馈和处理上做出改进。



项目健康度：

利用散点图展示各个项目的健康分数分布情况，并利用环形图展示出健康分数集中的分布区域。散点图能够有效展示项目健康分数的整体分布情况，而环形图则提供了更细化的视角，帮助我们识别出健康分数集中区域和潜在的风险项目。

通过明细表展示出处于异常健康状态的项目，并在旁边统计异常项目总数。该明细表将列出健康状态异常的项目，并提供其相关的详细信息，进一步帮助项目管理人员及时发现并解决项目健康状况的潜在问题。



1. **创新点**
2. 多维度评估体系创新：摒弃传统仅依靠星标数或 Fork 数的单一

评估模式，深度融合代码质量、社区活跃度、维护频率、用户参与度等多个维度，并依托专业的 Opendigger 指标体系进行综合考量。这一全面的评估方式打破了传统评估的片面性，能够从各个关键层面精准地衡量项目健康度，为项目评价提供了更客观、全面的视角，有效避免了因指标单一而导致的评估偏差。

1. 指标科学性强化创新：充分利用 Opendigger 经过广泛实践验

证的指标数据，并在此基础上进行细致的二次整合。通过对原始数据的深入挖掘和优化处理，确保了数据来源的专业性和可靠性，使得评估结果更能真实反映项目的核心健康状况，为项目决策提供坚实的数据支撑，增强了评估体系的可信度和权威性。

1. 可视化与分析功能创新：借助直观的大屏仪表盘和灵活的可交互

图表，打造了全新的可视化展示平台。用户不仅能够迅速获取各个项目的健康度概况，还能在需要时便捷地进行深度对比分析。这种创新的展示方式极大地提高了数据获取效率，帮助用户快速洞察项目间的差异和优势，从而显著提升决策效率，使项目评估过程更加高效、便捷。

1. 评分算法灵活拓展创新：采用先进的加权评分算法，可根据不同

的应用场景和用户需求，自由地自定义指标权重和组合方式。同时，具备良好的扩展性，能够轻松引入如异常检测等更多维度的指标，进一步提升评估体系的精准度和适应性。这种灵活性使得评估体系能够更好地满足多样化的评估需求，适应不断变化的项目环境。

5. 健康分数计算方法创新：在健康分数计算上，摒弃传统的算术平均法，采用独特的几何平均与线性缩放相结合的方式。几何平均对指标分数的变化具有高度敏感性，只要四个指标中的任意一个处于较低水平，整体健康分数就会被显著拉低，有效避免了个别指标不佳却被平均掩盖的问题。在此基础上进行线性缩放，将分数合理分布在 30 - 90 的区间内，进一步增强了分数的区分度和可读性，为项目健康度评估提供了更精确、更具区分性的结果，有力地辅助开发者和企业做出更科学、合理的决策，在项目健康度评估领域开辟了新的思路和方法。