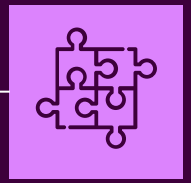
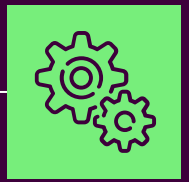
Análise de Repositórios do GitHub

Matheus Vinícius Mota Rodrigues
Gabriel Henrique Mota Rodrigues
João Queiroga

TABLE OF CONTENTS



01 Introdução



02 Hipóteses

03 Metodologia

04 Resultados

1. Introdução

Este trabalho tem como objetivo analisar as principais características de sistemas open-source populares hospedados no GitHub. Foram coletados dados de 1000 repositórios com maior número de estrelas, considerando aspectos como idade, contribuições externas, frequência de releases, atualizações recentes, linguagem primária e taxa de issues fechadas.

```
def run_query(variables):
    headers = {"Authorization": f"Bearer {GITHUB_TOKEN}"}
    response = requests.post(
        "https://api.github.com/graphql",
        json={"query": query, "variables": variables},
        headers=headers
    )
    if response.status_code != 200:
        raise Exception(f"Erro {response.status_code :
{response.text}")
    return response.json()
```

2. Hipóteses

Antes da análise, levantamos as seguintes hipóteses:

- H0: Repositórios mais antigos têm maior número de estrelas acumulados mas possuem menor taxa de crescimento recente.

- H1: Existe correlação entre número de estrelas e o número de contribuidores no repositório.

2. Hipóteses

HO - Repositórios mais antigos têm maior número de estrelas acumulados mas possuem menor taxa de crescimento recente.

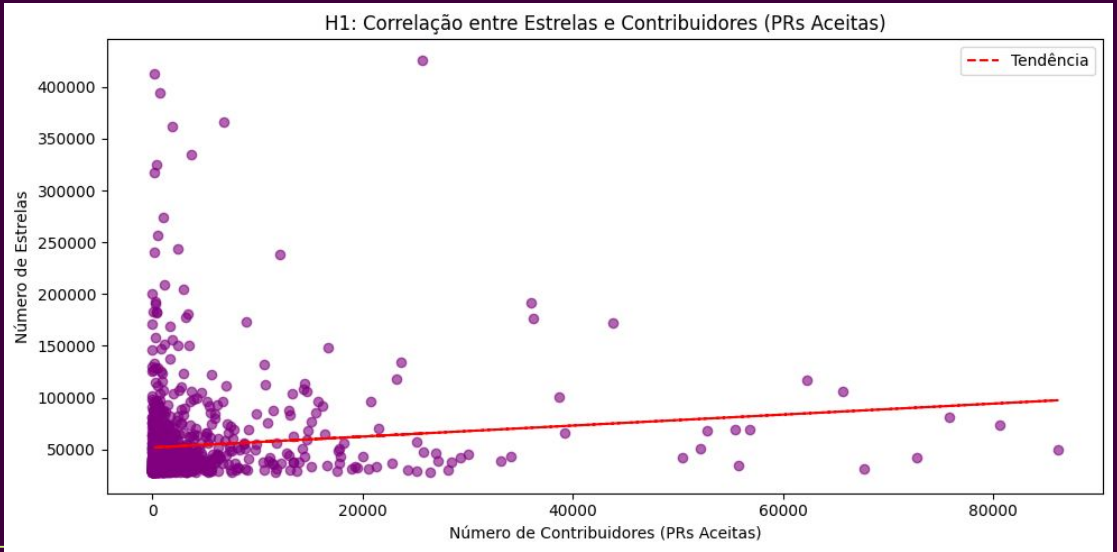
- Correlação idade x estrelas acumuladas
  Correlação calculada = r ≈ 0,066

- Correlação idade x taxa de crescimento
  Correlação calculada = r ≈ -0,501

- **A hipótese é suportada pelos dados**: projetos antigos tendem a acumular estrelas, mas o crescimento recente é mais forte em projetos novos.

2. Hipóteses

H1: Existe correlação entre número de estrelas e o número de contribuidores no repositório.



- Correlação:
- estrelas x contribuidores:

r ≈ 0,111

- **A hipótese é parcialmente confirmada:** há evidência de correlação, mas o efeito é fraco.

2. Metodologia

1. Coleta de dados:
    - Utilizamos a API GraphQL do GitHub.
    - Foram coletados 1000 repositórios mais populares em número de estrelas.
    - Para cada repositório, registramos
        - Nome
        - Dono
        - Quantidade de estrelas
        - Linguagem primária
        - Idade
        - Número de releases

2. Metodologia

2. Análise:

- Para métricas numéricas, calculamos mínimo, máximo, média e mediana.

- Salvamos os dados coletado em uma planilha de resumo

resumo_estatisticas

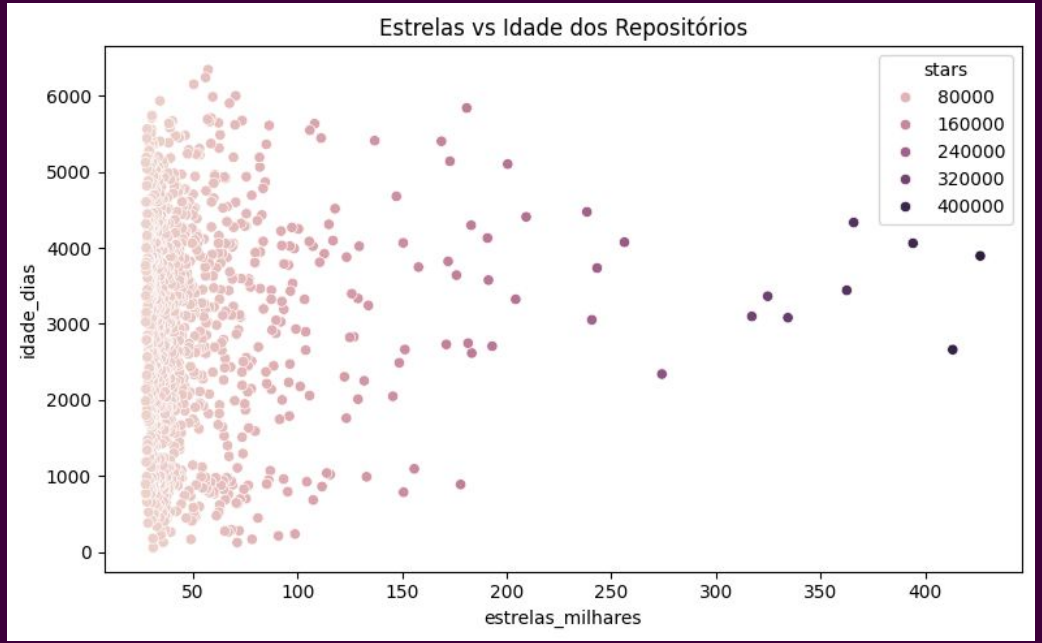
```
with open(RESUMO_FILE, "w", newline="", encoding="utf-8") as csvfile:
    fieldnames = ["RQ", "min", "max", "media", "mediana"]
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()
    writer.writerows(estatisticas)
```

4. Resultados

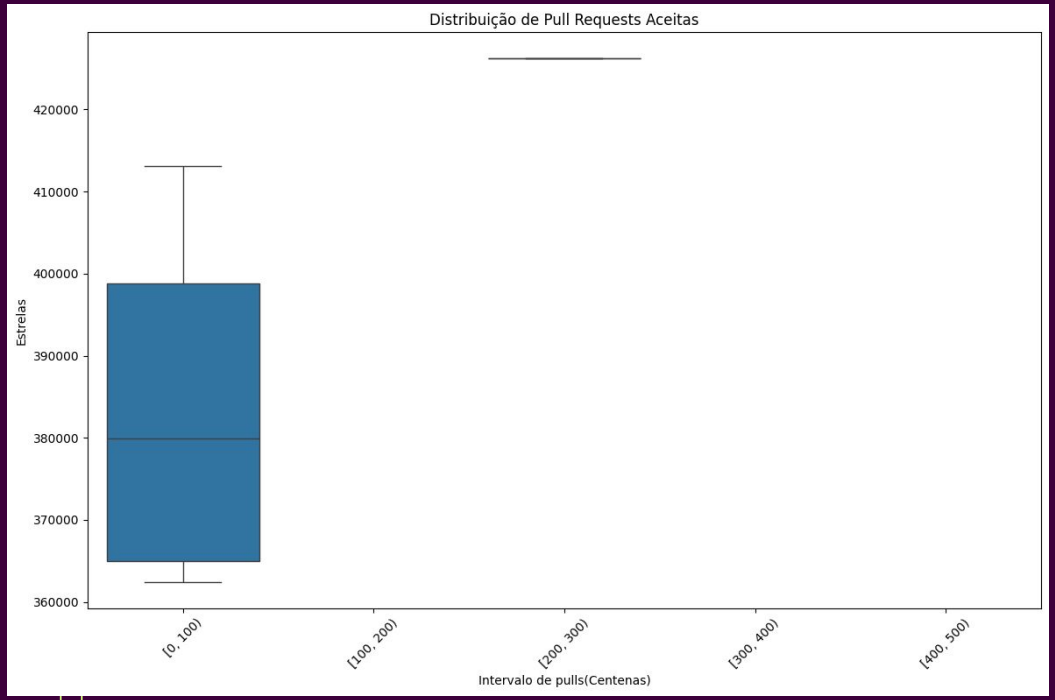
RQ01 – Sistemas populares são maduros/antigos?

- Métrica: idade do repositório (anos desde a criação).
- Mediana observada: ~ 3051 dias.

A correlação é de aproximadamente 0.066. Como esse valor está muito próximo de 0, não há correlação linear significativa entre ser mais antigo e ser mais popular.



4. Resultados



RQ02 – Recebem muita contribuição externa?

- Métrica: número de pull requests aceitas.
- Mediana observada: ~ 702 pull requests aceitas.

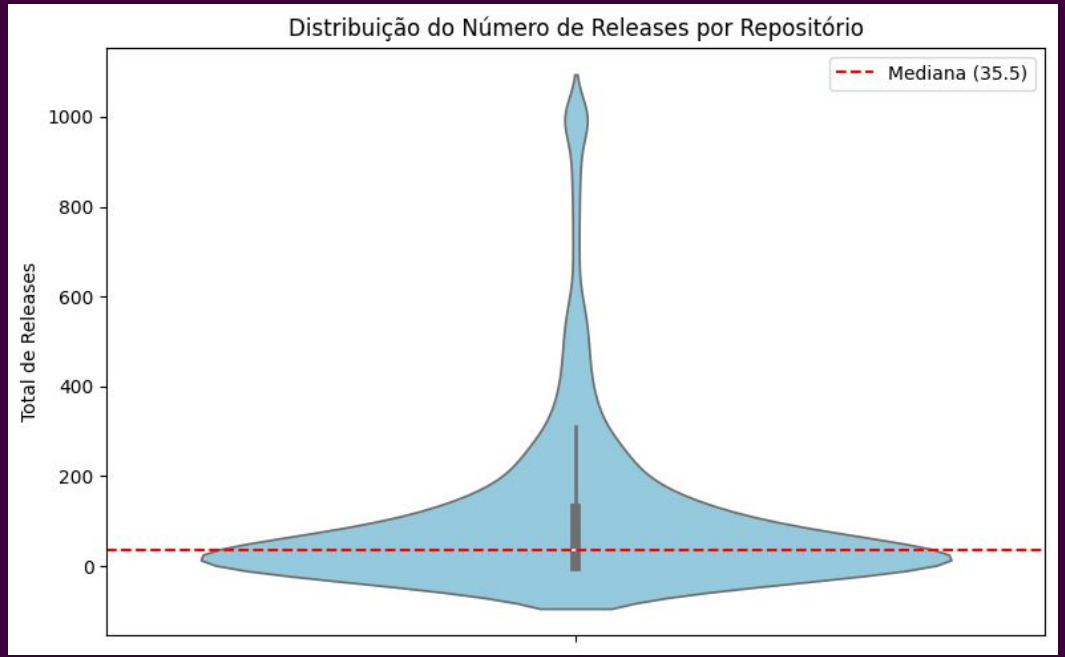
Também confirmada. O número elevado de pull requests aceitas indica grande colaboração externa.

4. Resultados

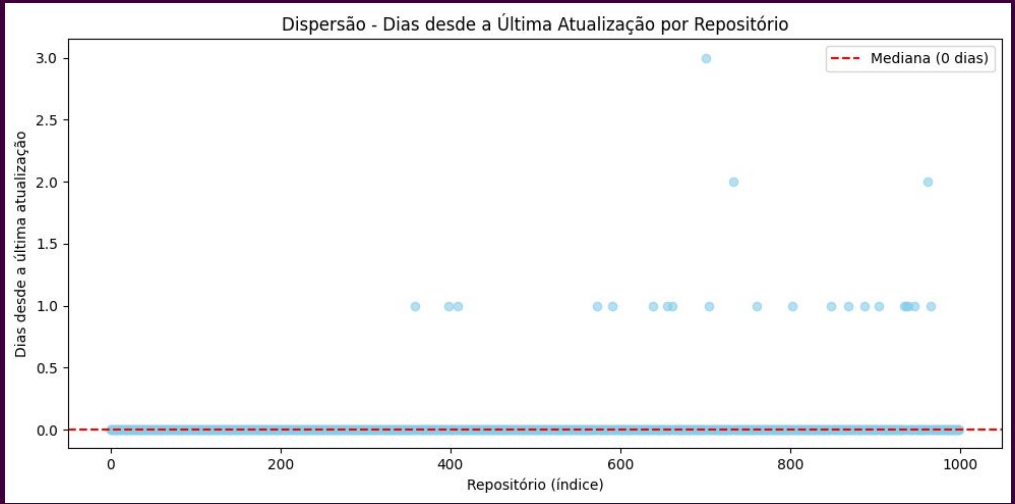
RQ03 – Lançam releases com frequência?

- Métrica: número de releases.
- Mediana observada: ~ 35.5 releases por repositório.

Confirmada parcialmente. Alguns sistemas realmente possuem releases frequentes (frameworks), mas outros (repositórios de aprendizado) quase não lançam releases.



4. Resultados



RQ04 – São atualizados com frequência?

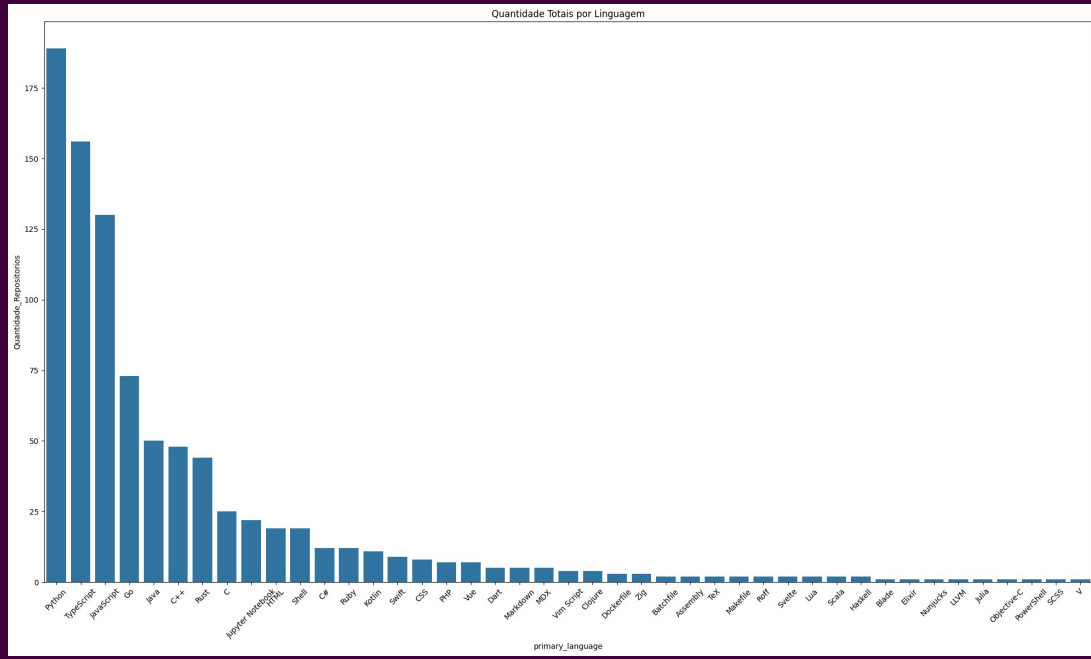
- Métrica: tempo desde a última atualização.
- Mediana observada: 0 dias desde a última atualização.

Confirmada. A maioria dos repositórios possui atividade recente (últimos dias/semanas).

4. Resultados

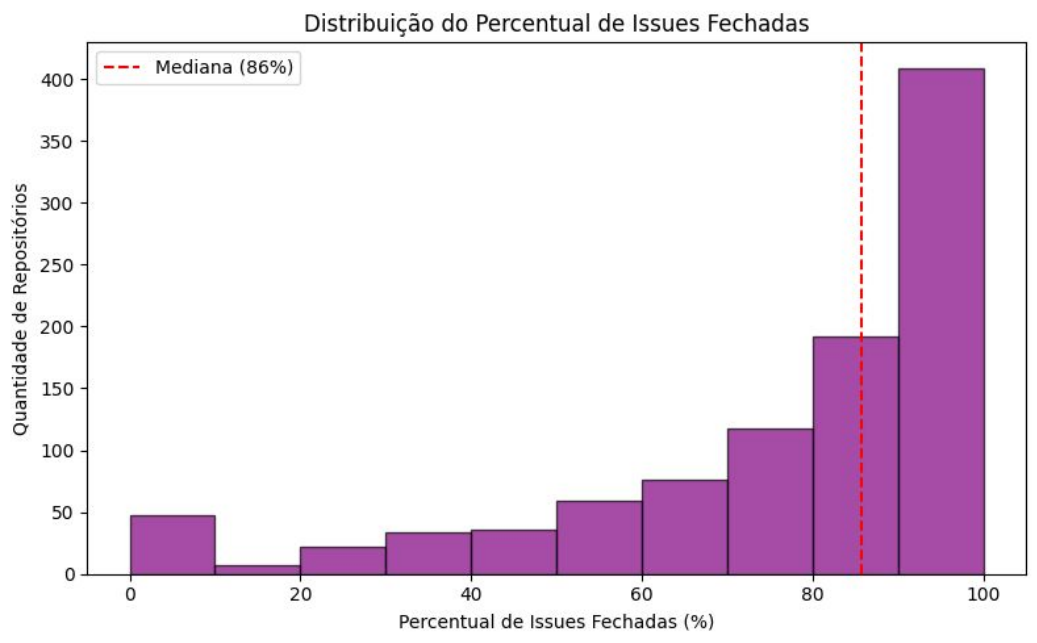
RQ05 – Linguagens mais utilizadas

- Métrica: linguagem primária.
- Distribuição (top 5):
  - Python - 21.07%
  - TypeScript - 17.39%
  - JavaScript - 14.49%
  - Go - 8.14%
  - Java - 5.57%



Confirmada. JavaScript, Python e TypeScript aparecem como as linguagens mais comuns.

4. Resultados



RQ06 – Percentual de issues fechadas

- Métrica: closed issues / total issues.
- Mediana observada: 85% de issues fechadas.

Confirmada parcialmente. Embora a mediana de 86% seja alta, alguns projetos apresentam grande número de issues abertas, podendo refletir dificuldades de manutenção.

Conclusão

Em suma, os achados reforçam que o sucesso e a longevidade de projetos open-source estão associados a múltiplos fatores, incluindo idade, colaboração da comunidade, frequência de atualizações e capacidade de manter uma base de usuários engajada. Estudos futuros podem aprofundar essa análise, investigando a relação entre diferentes modelos de governança, financiamento e sustentabilidade dos projetos.