Relatório - Lab01

Eduardo Bandeira de Melo Guimarães

1. .

   A. **Hipótese:** Os repositórios mais populares do GitHub (com as maiores quantidades de estrelas) são maduros/antigos.

   B. **Justificativa da hipótese:** Acredito que apenas a qualidade de um projeto não basta para que este se torne amplamente conhecido, utilizado e aprovado. Acredito que o tempo também é crucial por uma série de motivos, como por exemplo: é ao longo do tempo que problemas e vulnerabilidades vão sendo descobertas e corrigidas; o tempo muitas vezes filtra pessoas mal-intencionadas e também leva-se tempo para que a informação sobre a existência de determinado projeto se espalhe pela comunidade. Todos esses aspectos são verdadeiros para vários outros aspectos da atividade humana, e acredito que estes se mantêm válidos para projetos open-source.

   C. Metodologia: Para responder a esta questão, foi realizada uma consulta à API do GitHub usando GraphQL, através de um script escrito em Python. Foram analisados os 1000 repositórios mais populares da plataforma, ou seja, com as maiores quantidades de estrelas atribuídas. Para a filtragem desta consulta, foi utilizado o campo “createdAt”.

   D. Resultados obtidos: O resultado encontrado mostrou que a mediana da “idade” dos 100 repositórios mais populares é de 3.025 dias, ou seja, aproximadamente **8 anos e 3 meses.** O conceito de maturidade de repositórios é bastante relativo, mas, na minha opinião, pode-se dizer que 8 anos de projeto representa um elevado grau de maturidade.

   E. Discussão sobre resultados esperados e obtidos: O resultado obtido confirma a hipótese levantada.

2. .

   A. **Hipótese:** Projetos populares recebem muita contribuição externa.

   B. **Justificativa da hipótese:** No contexto específico de projetos de código aberto, entendo que um grande fator determinante de seu sucesso é a quantidade de colaboradores, vide projetos amplamente utilizados e respeitados como o Linux e o Bitcoin. Além disso, é muito lógico pensar que a popularidade de um projeto e a quantidade de colaboradores são duas variáveis diretamente proporcionais, ou seja, projetos com muitos colaboradores são mais divulgados e comentados, da mesma forma que a própria popularidade também traz novos colaboradores, o que faz dessa relação uma espiral retroalimentada.

   C. Metodologia: Para responder a esta questão, foi realizada uma consulta à API do GitHub usando GraphQL, através de um script escrito em Python. Foram analisados os 1000 repositórios mais populares da plataforma, ou seja, com as maiores quantidades de estrelas atribuídas. Para a filtragem desta consulta, foi utilizado o campo “pullRequests”, com o modificador “states: MERGED”, juntamente ao subcampo “totalCount”.

   D. Resultados obtidos: O resultado encontrado foi uma **mediana de 620 pull requests aceitos**. Trata-se de um valor bastante expressivo.

   E. Discussão sobre resultados esperados e obtidos: O resultado obtido confirma a hipótese levantada.

3. .

   A. **Hipótese:** Projetos populares lançam releases com frequência.

   B. **Justificativa da hipótese:** O uso amplo de determinada tecnologia leva inevitavelmente à descoberta de falhas e pontos de melhoria, o que consequentemente levaria a uma necessidade de lançamentos frequentes de releases que incorporassem essas melhorias.

   C. Metodologia: Para responder a esta questão, foi realizada uma consulta à API do GitHub usando GraphQL, através de um script escrito em Python. Foram analisados os 1000 repositórios mais populares da plataforma, ou seja, com as maiores quantidades de estrelas atribuídas. Para a filtragem desta consulta, foi utilizado o campo “releases”, juntamente ao subcampo “totalCount”.

   D. Resultados obtidos: O resultado encontrado foi uma mediana de **33 releases** para os 1000 repositórios mais populares do GitHub. Trata-se de um valor bastante reduzido.

   E. Discussão sobre resultados esperados e obtidos: O resultado encontrado refuta a hipótese inicialmente levantada. Diversos repositórios analisados nunca lançaram nenhuma release. Um motivo que pode explicar esse comportamento é o de que muitos projetos podem não “oficializar” as mudanças feitas como releases.

4. .

   A. **Hipótese:** Não existe uma relação direta entre a popularidade de um repositório e a proximidade da última atualização deste.

   B. **Justificativa da hipótese:** Não nego essa relação, apenas acredito que a métrica usada para avaliar popularidade pode ser falha neste aspecto. Na minha opinião, o número de estrelas de um projeto não é uma métrica com um bom atributo de sensibilidade, ou seja, ela não responde em um tempo tão satisfatório às alterações negativas de popularidade. Assim, projetos que foram sendo abandonados com o tempo podem continuar com muitas estrelas pelos mais diversos motivos, como simplesmente porque os usuários se esqueceram de retirar a estrela, não ligaram para isso, perderam suas contas, abandonam o GitHub, etc. Acredito que repositórios que já foram muito populares no passado e não o são mais podem continuar com muitas estrelas e, assim, aqueles considerados entre os mais populares podem ter um tempo muito grande desde a última atualização.

   C. Metodologia: Para responder a esta questão, foi realizada uma consulta à API do GitHub usando GraphQL, através de um script escrito em Python. Foram analisados os 1000 repositórios mais populares da plataforma, ou seja, com as maiores quantidades de estrelas atribuídas. Para a filtragem desta consulta, foi utilizado o campo “latestRelease”, com o sub-campo “publishedAt” dos repositórios mais populares. Para o cálculo da métrica foram considerados apenas os projetos que possuem dados sobre a última release. Ou seja, repositórios onde este valor era nulo foram descartados da análise. Assim, é importante destacar que **dos 1000 repositórios mais populares do GitHub, apenas 681 continham dados disponíveis sobre o último lançamento**.

   D. Resultados obtidos: O resultado encontrado foi uma **mediana de 43 dias desde o último lançamento de uma release.** Releases com 42 dias de lançamento podem ser consideradas como consideravelmente recentes.

   E. Discussão sobre resultados esperados e obtidos: Os resultados obtidos refutam a hipótese levantada. Provavelmente, a hipótese levantada de que a quantidade de estrelas não é uma métrica com um bom atributo de “sensibilidade” esteja equivocada.

Provavelmente, repositórios abandonados perdem relevância neste quesito em um tempo hábil.

5.

A. **Hipótese:** Os projetos mais populares são escritos nas linguagens mais populares.

B. **Justificativa da hipótese:** Projetos que utilizam linguagens populares são projetos que provavelmente terão mais usuários e mais colaboradores, o que aumenta a probabilidade de que o projeto se torne popular.

C. Metodologia: Para responder a esta questão, foi realizada uma consulta à API do GitHub usando GraphQL, através de um script escrito em Python. Foram analisados os 1000 repositórios mais populares da plataforma, ou seja, com as maiores quantidades de estrelas atribuídas. Para a filtragem desta consulta, foi utilizado o campo "primaryLanguage", com o sub-campo "name" dos repositórios mais populares.

D. Resultados obtidos: Descobriu-se que **79% dos 1000 repositórios mais populares do GitHub apresentam como linguagem principal alguma das 10 linguagens mais populares do GitHub.**

E. Discussão sobre resultados esperados e obtidos: O resultado obtido confirma a hipótese levantada.

6. .

A. Hipótese: Projetos populares possuem um alto percentual de issues fechadas.

B. Justificativa da hipótese: Projetos populares tipicamente são aqueles que apresentam muitos colaboradores, o que, provavelmente, contribui positivamente para o percentual de issues fechadas.

C. Metodologia: Para responder a esta questão, foi realizada uma consulta à API do GitHub usando GraphQL, através de um script escrito em Python. Foram analisados os 1000 repositórios mais populares da plataforma, ou seja, com as maiores quantidades de estrelas atribuídas. Para a filtragem desta consulta, foi utilizado o campo "issues" com o modificador "states: CLOSED", juntamente ao subcampo "totalCount"; além do campo "issues" com o subcampo "totalCount".

D. Resultados obtidos: A mediana da razão entre issues fechadas e issues totais para os 1000 repositórios mais populares do GitHub foi de **0,86574 (86,57%)**. Trata-se de um valor bastante expressivo.

E. Discussão sobre resultados esperados e obtidos: O resultado obtido confirma a hipótese levantada.