Exploring Technology Landscape Via GitHub Data

Anh Tran, Ha Nguyen, Ha Le, Khoi Nguyen, Thanh Nguyen, Tuan Pham

Contents

1 Introduction

1.1 Overview

Technology is changing quickly that requires us to learn and adapt everyday. What are the dorminant technologies? What factors make some tools more important than others? How do we know what to learn in this world of abundant opportunities and limited resources? In a effort to answer such questions, we leveraged statistical tools and the GitHub API to analyze the data from the open-source projects. Our results show the dorminant programming languages, as well as factors that affect the popularity of a GitHub repository. We published our source code at `https://github.com/npnkhoi/github-data-analysis`.

1.2 Background: GitHub Repositories

Our dataset comes from GitHub, an online hub to share, store code, and collaborate on coding projects. Each project on GitHub is organized in a *repository* (henceforth "repo"). There are 200 million repositories on GitHub as of June 2022[1].

As a large portion of repositories on GitHub is publicly available, the public audience can interact with the repositories by giving stars (i.e. appreciation), watching (i.e. subscribing), or forking (i.e. cloning the project to develop parallelly). A repository can also have issues that are raised by anyone, which may be resolved by the maintainers of the repository.

In this paper, we record such aspects of a repository by the following statistics:

Attribute name	Example
Full name	`recloudstream/cloudstream`
Number of stars	`1623`
Number of forks	`126`
Number of open (unresolved) issues	`185`
Main programming language	`Kotlin`
Time of creation	`2022-08-04T08:42:38Z`
Time of the last update	`2022-12-03T23:06:29Z`
License key	`gpl-3.0`
List of topics (or tags)	`'android, anime, media-center, [...]'`

Table 1: Attributes of each repository

1.3 Research Questions

We aim to answer the following questions

1. RQ1: What is the overall characteristics of the dataset from GitHub?
2. RQ2: What is the distribution of programming languages used?
3. RQ3: Do license key and language have any effect on the number of stars?

[1] `https://en.wikipedia.org/wiki/GitHub`

4. RQ4: Is there any significant difference in the number of open issues between the popular and unpopular projects?

5. RQ5: Is there any significant difference in the number of stars between repositories with topics keywords specified and those without?

2 Methods

2.1 Data Collection

In order to answer our research questions, we need to collect a sufficient amount of data. Therefore, we aimed to scrap 2000 GitHub repos and recorded their information by the following procedure:

- **Group sampling**: Randomly select a date between the date Github was founded (April 1, 2008) and today, then pick 100 most starred repos in that date.
- **Data scrapping**: For each repo, we record the necessary information (as shown in Table 1).

Our script makes use of GitHub Search API[2] to automatically scrap data from GitHub's server. As a result, we have 1993 repos in our dataset.

2.2 Statistical Analysis

We made use of many tools to answer the research questions as well as to explore their usage. We describe them briefly below.

2.2.1 Descriptive Statistics

For numerical data (e.g. number of stars and forks), we used measures of central tendency, measures of variability (or spread), and frequency distribution. For categorical data (e.g., topics, descriptions, license key) we used methods like repetition counting and manual classification. Below, we list the detailed techniques used.

2.2.2 Shapiro-Wilk Sest for Normality

This is a test of normality in frequentist statistics [3] With an α level of 0.05, a data set with a p-value of less than .05 rejects the null hypothesis that the data are from a normally distributed population.

2.2.3 ANOVA Hypothesis Testing

The analysis of variance or ANOVA is a statistical inference test that lets you compare multiple groups at the same time. There is one treatment or grouping factor with $k > 2$ levels and we wish to compare the means across the different categories of this factor.

There are three main assumptions in ANOVA hypothesis testing. Firstly, the responses for each factor level have a normal population distribution. Secondly, these distributions have the same variance. Finally, the data are independent.

ANOVA assumes the means of all groups are the same, calculates how much the actual means deviate from the assumption and reports it as the F-test score. A larger score means there is a

[2] `https://docs.github.com/en/rest/search`

[3] `https://en.wikipedia.org/wiki/Shapiro%E2%80%93Wilk_test`

larger difference between the means. Finally, a P-value tells how statistically significant is our calculated score value.

2.2.4 Student's T-test for Sample Mean

A two-sample z-test is used to examine equality between two sample means, assuming in each population is approximately normally distributed. Null hypothesis of mean equality (two-tail) rejection is based on t-score with corresponding p-value.

2.2.5 Multinomial Distribution

This is a common probability model[4] for categorical data. It is a generalization of the binomial distribution to more than two possible categories of the outcome. In an independent sequence of n trials, each of which has probability p_i of resulting in an outcome in category i, $1 \leq ilek$, and $p_1 + p_2 \cdot +p_k = 100$, the numbers $X_1, X_2, \ldots, X_k$ of outcomes in each of the k categories have a multinomial joint probability distribution: If $n_1 + n_2 + \cdots + n_k = n$,

$$P\left(X_1 = n_1, X_2 = n_2, \ldots, X_k = n_k\right) = \frac{p_1^{n_1} p_2^{n_2} \cdots p_k^{n_k}}{n_1! n_2! \cdot n_k!}.$$

2.2.6 Chi-square Test for Goodness of Fit

Let

$$\chi^2 = \frac{\left(X_1 - np_1\right)^2}{np_1} + \frac{\left(X_2 - np_2\right)^2}{np_2} + \cdots + \frac{\left(X_k - np_k\right)^2}{np_k}.$$

If $n \cdot p_i > 10$ for every $i = \overline{1, k}$, the probability histogram of chi-squared when the null hypothesis is true can be approximated accurately by the chi-square curve with $k - 1$ degrees of freedom. The null hypothesis is rejected if chi-squared $> \chi_{k-1,1-\alpha}$, where α is the significant level.

2.2.7 Linear Regression

Let $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ be n pairs of observations such that y_i is an observed value of the random variable Y_i. Assuming there are exist constants β_0 and β_1 such that:

$$Y_i = \beta_0 + \beta_1 x_i + E_i$$

where $E_1, E_2, \ldots, E_n$ are independent, follow a normal distribution with mean 0 and variance σ^2. The null hypothesis is that there is no significant linear relationship ($\beta_1 = 0$) and the alternative hypothesis is there is a significant relationship ($\beta_1 \neq 0$). This hypothesis testing using the F-distribution.

[4] https://www.stat.berkeley.edu/~stark/SticiGui/Text/chiSquare.htm

3 Results

We will answer each RQ in the section below.

3.1 RQ1: What are the overall characteristics of the dataset from GitHub?

As each repo in our data contains two kinds of attributes, numerical and categorial, we present such descriptive statistics separately.

3.1.1 Overview of Numerical Data

We consider the 3 numerical attributes of each repo, i.e. `num_stars`, `num_forks`, and `num_open_issues`.

Distribution We were first concerned with the distribution of these attributes because that would inform the tools we could use. As shown in histograms and box plots (Figure 1), the three distributions are highly right-skewed.

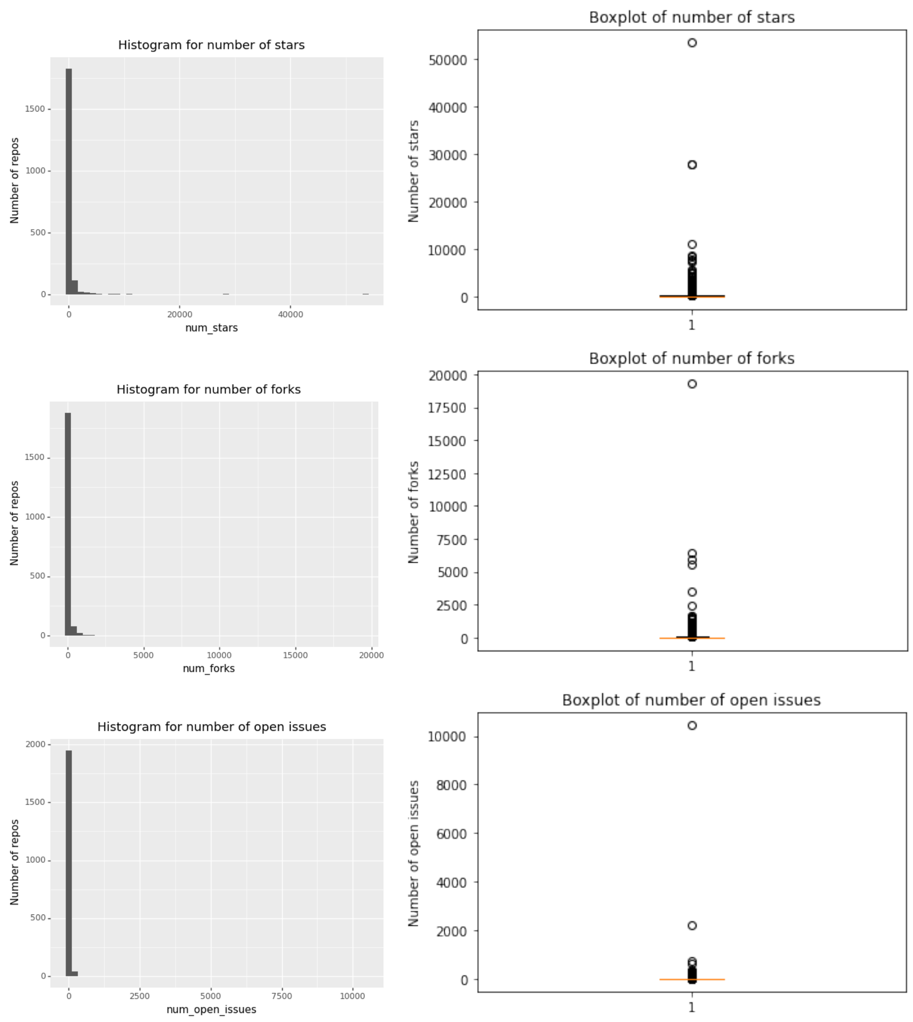

Figure 1: Histograms and box plots for data of the number of stars, forks, and open issues. The black region is the outliers data, while 50 % of data focuses on one region: the red/black line, which represents the lower whisker and upper whisker.

We further tested their normality using the Shapiro-Wilk test. Based on the results in Table 2, we can say that the distributions of the three variables above are not normal.

Name of variables	Test statistic	p-value
Number of stars	0.11	0.0
Number of forks	0.08	0.0
Number of open issues	0.03	0.0

Table 2: Result of the Shapiro-Wilk test for normality

Central tendency

	number of stars	number of forks	number of open issues
count	1993	1993	1993
mean	279.110888	73.470647	17.068741
std	1636.886047	517.645541	241.766542
min	1	0	0
25%	10	2	0
50%	57	11	1
75%	153	38	6
max	53545	19281	10441

Table 3: Descriptive statistics for the numerical feature of projects

Table 3 shows the central tendency based on the calculation of count, mean, standard deviation, first quartile, third quartile, max, and min. We can see that 75% of each data is quite small compared to the max value. That's why the standard deviation is significantly big compared to other values.

Dispersion
For the shake of completeness, below are the 95 % confidence interval for the mean of each population:

- Number of stars: $(207.2, 351.0)$
- Number of forks : $(50.7, 96.2)$
- Number of open issues : $(6.4, 27.7)$

3.1.2 Overview of Categorial Data

a. Language

Figure 2 shows the distribution of the top 15 languages that appeared in our data. Python, Javascript, and Ruby are the most used languages on GitHub, with more than double the repos compared to the fourth language (Java).

b. Topics and Descriptions

In this part, we explored the qualitative data of topics and descriptions.

For the descriptions in the repos, we draw their world cloud (Figure 3) and gain 4 insights. Firstly, Python is a hot language. It may be because Python is useful in all industries. This result matches the bar chart of the number of the main language used. Next, Android appears

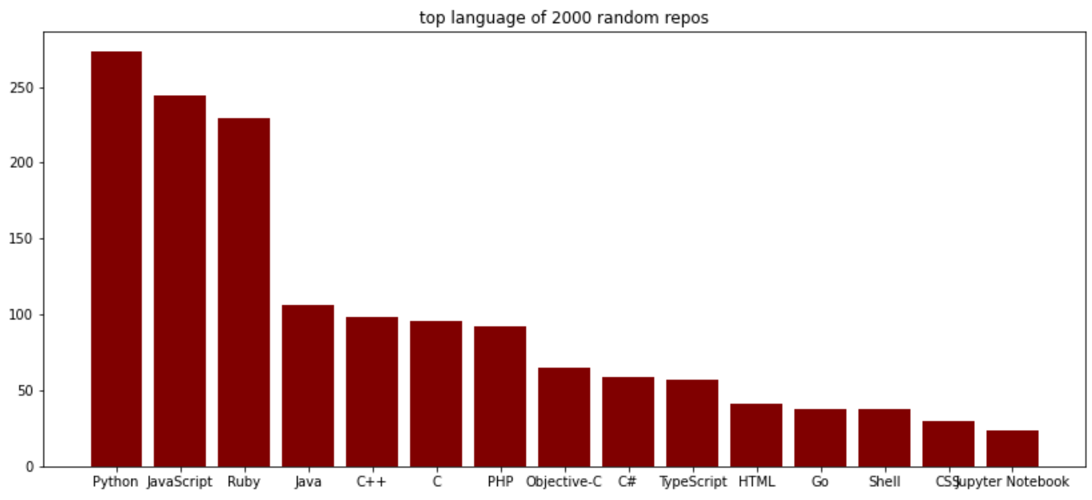

Figure 2: Language frequency

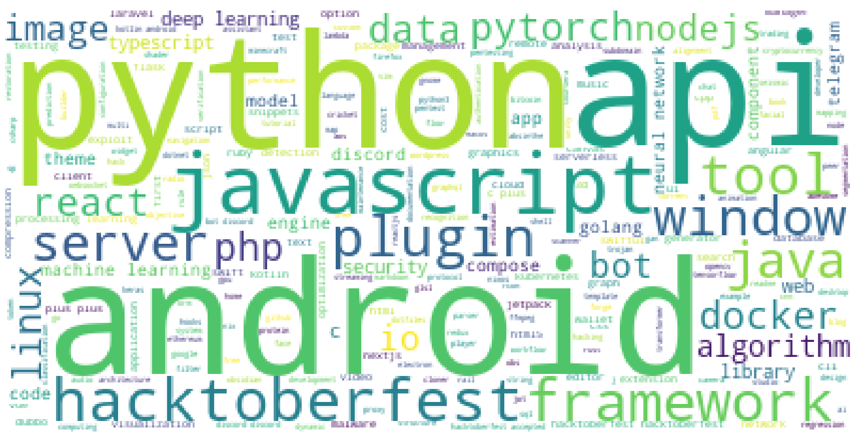

Figure 3: Word cloud of topics

to be more popular than iOS. That may be due to its open-source nature. Finally, API seems to be hot as well as it is definitely the backbone of the Web. Finally, “Hacktoberfest” is a strange word that appears frequently in our data. It is actually a worldwide competition for beginners.

In addition to project description, repositories are commonly labelled by contributors by topic keywords. Such keywords inform users an overview into many aspects of the project (purposes, technology, techniques, programming languages...).

We classified the repos into relevant technology domains. Classification process is based on automatic keyword matching as well as manual inspections. Repos with too few or non-specific keywords are labeled “Other”. Repos with unspecified topics (tags) are excluded from the labeling process. Results are shown in Table 4. The Machine Learning/Artificial Intelligence/Data Science domain receives the most attention from the community, followed by Web/Software Development, Code Editor/Command-line Interface tools, and Plugins/Bots/Extension.

Topic	Project count	Keywords
Web/Software Development	146	webpack, web-framework, devops, etc.
ML/AI/Data Science	73	machine-learning, neural-network, tensorflow, etc.
Editor/CLI/OS Tools	62	cli, command-line, vim, neovim, etc.
Plugins/Bots/Extensions	48	chatbot, plugins, extension, api-client, ...
Security	29	firewall, http-proxy, proxy, vpn, security, ..
Web3	23	crypto, blockchain, discord, web3, ..
Graphics	22	graphic, opengl, unity, unity3d, 3d-graphics, svg, ...
Robotics/Engineering	16	reverse-engineering, robotics, robot, ...
Hacktober	16	hacktober, hacktoberfest, challenge
Resources	9	tutorial, awesome, awesomelist, edu, etc.
Other	40	github, opensource, beginner, ...
Total	485	

Table 4: List of domains

3.2 RQ2: What is the distribution of programming languages used?

In our data, 1770 repos have information about the main language, with 95 distinct values. We excluded projects that do not have this field, as well as those that contain an invalid programming language. That leaves us with 68 programming languages in 1552 projects. We plot the frequencies of the top most used 10 languages with the highest frequency, along with the rest (called "Others") in Figure 4.

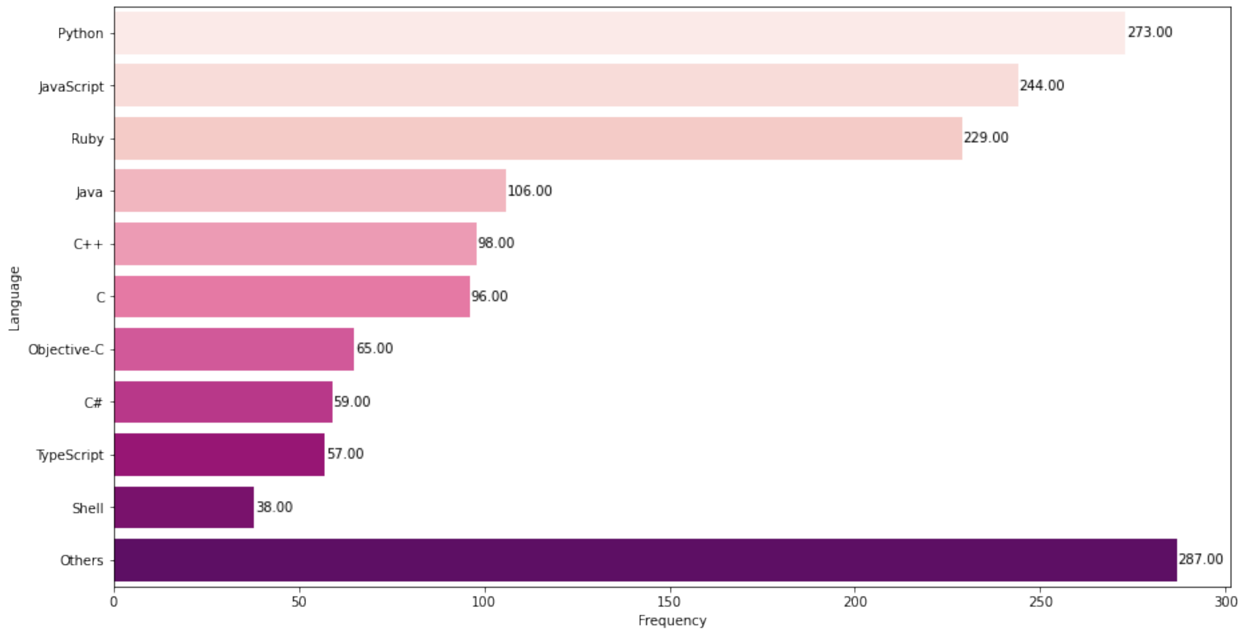

Figure 4: Main language frequencies

We can see that the top 10 most used programming languages are: Python, JavaScript, Ruby, Java, C++, C, Objective-C, C #, TypeScript, and Shell. One can say that their percentages and the group of remaining languages could be $15.6\%, 15.6\%, 15.6\%, 7.8\%, 7.8\%, 7.8\%, 3.9\%, 3.9\%, 3.9\%, 2.6\%, 15.6\%$. Looking at this distribution, we guess the true proportion of frequencies among the top 10 programming languages is $1 : 1 : 1 : \frac{1}{2} : \frac{1}{2} : \frac{1}{2} : \frac{1}{4} : \frac{1}{4} : \frac{1}{4} : \frac{1}{6} : 1$. Is the true ratio indeed these numbers? We used the Chi-square test (right-tail) to validate, with the following hypotheses:

- *Null hypothesis*: The proportion of programming languages used is $1:1:1:\frac{1}{2}:\frac{1}{2}:\frac{1}{2}:\frac{1}{4}:\frac{1}{4}:\frac{1}{4}:\frac{1}{6}:1$.
- *Alternative hypothesis*: The proportion of programming languages used is not $1:1:1:\frac{1}{2}:\frac{1}{2}:\frac{1}{2}:\frac{1}{4}:\frac{1}{4}:\frac{1}{4}:\frac{1}{6}:1$.

The test statistic for our hypothesis is approx 25.25, while the Chi-square value at the degree of freedom 1551 is 1643.73, which is larger than 25.25. Thus, we cannot reject the null hypothesis. Therefore, one cannot reject the ratio above as not real.

Meanwhile, the fact that the group of languages that are not in the top 10 group accounts for a significant percentage (equal to the language with the largest percentage) shows the diversity of programming languages. The popularity of each language in this group is insignificant compared to the languages in the top 10, however, when combined, they account for a large proportion.

3.3 RQ3: Do license key and language have any effect on the number of stars?

We used ANOVA to know how two seemingly independent variables language and license key, individually and in combination, affect the dependent variable number of stars. To make sure all the pairs of languages and license keys exist, we only use the repositories which are the top 5 common languages, and the top three license keys (gpl, mit, and apache). Then, with the remaining 360 repositories, these are 3 hypotheses to test:

- *Null hypothesis 1*: The number of stars is related not to the combination of language type and license key.
- *Null hypothesis 2*: The number of stars is not related to the main language
- *Null hypothesis 3*: The number of stars is not related to the license key

	sum sq	df	F	$PR(> F)$
C(main language)	1.79e+07	4.0	1.46	0.21
C(license key)	3.5+06	3.0	0.38	0.77
C(license key):C(main language)	2.36+07	12.0	0.64	0.81
Residual	1.046690e+09	340.0	NaN	NaN

Table 5: ANOVA result when testing the effects of language and license key on the number of stars

Table 5 shows that the number of stars isn't related to the type of language or license key or the combination of language and license key.

3.4 RQ4: Is there a relationship between the number of open issues and the number of stars?

A project is considered popular if its number of stars is greater than 100, otherwise, it is considered unpopular. [5]. In GitHub, the metric used to evaluate the popularity of a project

[5]https://ieeexplore.ieee.org/document/8754436

is the number of stars. The motivation to give the star a project is to show appreciation or to bookmark the project.

As shown in Section 3.1.1, the distribution of the number of stars is very right-skewed with a lot of outliers. Therefore, we only keep 98% quantiles of the data. Also, due to the same reason, we consider the log value of the number of stars instead of the raw values. Figure 5 shows the scatter plot of these two variables.

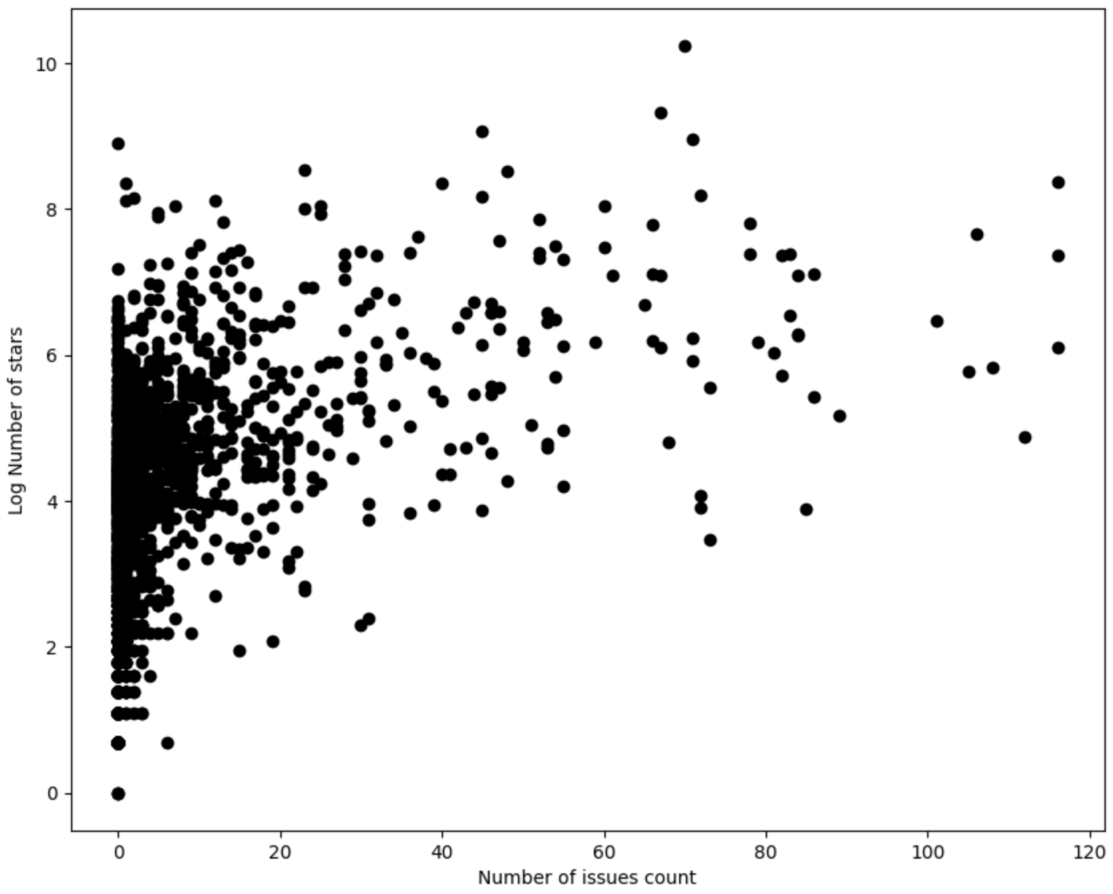

Figure 5: Scatter plot of the number of issues and the log number of stars

As these two variables appear to not be linearly correlated, we used quadratic regression to find any correlation. The final model gives an R-squared value of ≈ 0.29. Therefore, there is not any evidence to suggest a relationship between the number of open issues and the popularity of the projects.

3.5 RQ5: Is there any significant difference in the number of stars between repositories with topics keywords specified and those without?

A number of starred watchers (aka. stars) is a common metric to estimate the quality of GitHub repositories in terms of popularity [2].

Shapiro-Wilk testings for normality for star number distribution in all topic domain yield p-value smaller than 0.05, which implies non-normal distribution from the data. And thus, we do not proceed with hypothesis testing. However, we would like to see if having topics labeled any help with a project's popularity

Of 1354 repositories without any topic keywords specified and 426 those with, whose star numbers mean respectively 49.4224, 197.105, variance 3395.7, 34442.2, we randomly select 300 repositories on each group and perform two-tail student's t-test for sample means. For sample size 300, we assume an approximation of normality by the law of large numbers. T-test result $(\text{statistic} = -13.949, \text{p-value} = 3.45158\,\mathrm{e}\,{-39} < 0.05)$ suggests a significant difference in mean star number of repositories with and without topic specification.

Overall, repositories with topics specified may have more stars in average, and possible, more popular than those does not. This can be explained by the fact that topics keyword are short, highlighted as a repository that appears on the search results, and that topic tags also help users quickly identify and spot the project.

4 Discussions

4.1 Result Summary

From our dataset, we can say that the popularity metrics on Github has a right-skew distribution, i.e. only a few outstanding repositories become popular. Also, “Python”, “Android”, and “Hacktobefest” are the most popular keywords. We also conduct some predictions for the language and topics’ distribution in order to see their tendency.

Among variables that may influence the popularity of a project (measured by the number of stars), whether the project has a keyword has a strong effect on its popularity. Neither the number of open issues, the license key, nor the language used affect the number of stars.

4.2 Threats to Validity

There are some points worth noticing about the validity of our findings.

Firstly, the proportion of the programming languages used that we propose may change. In fact, the number of programming languages that have been released so far is estimated at 250-2500 languages. So, even though we have collected data with a size of 2000, this number is still not too large when compared to the number of programming languages. This proportion may also change because of the fact that data-related problems are becoming more and more popular and widespread, appearing across all industries. Therefore, Python - the preferred programming language for handling Data-related problems is believed to become more and more popular.

Secondly, in the analysis process, there are some repositories that are tagged by many topics. But some of them exists with no tags, which will be dropped. That leaves us with less data, which may result in less reliable conclusions.

5 Conclusion

In this project, we have applied both descriptive and inferential statistics to analyze a dataset we mined from GitHub. This experiences teaches us multiple things about data analysis.

Firstly, data sampling method is crucial. Initially, we scrap 500 most starred repos on GitHub because that sounded natural to us. However, after reflecting on the RQs, we realized that we need to sample randomly over all GitHub repositories. And it turned out to be a challenging task as GitHub does not expose any APIs to randomly pick a repo from it database. We eventually overcame that by randomly choosing a date and collect repos from that date, as this operation is supported by GitHub API.

Secondly, to make the out of the data, it is important to ask the right question. There were many times where we were attempted to apply a particular method to see its result without any particular goal. However, we quickly agreed afterwards that it is best to follow the original RQs closely, and reason about the tools we have to use the right one.

Finally, GitHub is a fun place to explore about technologies. It is complex for beginners (like non-CS members in our group), but it quickly appears to be a lively social media platform for technology-lovers. It is an open place for learning and building in this ever-evolving world.

References

[1] *A large scale study of programming languages and code quality in github.* Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. https://doi.org/10.1145/2635868.2635922

[2] Jarczyk, O., Gruszka, B., Jaroszewicz, S., Bukowski, L., Wierzbicki, A. (2014) *Github projects. quality analysis of open-source software., in International Conference on Social Informatics (pp. 80-94) Springer, Cham.*

[3] Stephen, K. (2015). "Chapter 12: Correlation and Linear Regression." *Introductory statistic, 2th edition (pp. 575-576)*