# GitHub Analysis

Aishwarya Ramachandrappa
Krupa Mavani
Sahana Chambakur Sreenivasulu

December 14, 2018

## 1 INTRODUCTION

GitHub is the most popular web-based version control platform which offers several features such as code management, easy collaboration and social coding. It was first launched in 2008 and has constantly been one of the best version control platforms. Though many such version control systems exists, the social features of GitHub make it most popular and widely used. Some of the features are following a user, raising issues in a repository, commenting on a repository, watch a repository etc. As of 2018, GitHub has over 28 Million users and 57 Million repositories.

The main aim of this project is to use this platform to understand the evolution in technology and be well informed about the current trend. We also try to establish a relationship between the users to form a network. GitHub API - *PyGitHub* [1] will be used to extract data and perform analysis. We have used the information available and take advantage of GitHub's main features - commits, pull requests, and issues to perform analysis and create visualizations.

## 2 DESIGN

### 2.1 OVERVIEW OF THE APPLICATION

The application takes different inputs based on the type of analysis to be done. Users can provide input such as *User Name, Repository Name or Language*. Based on the input various

charts are displayed which help us understand the data better. We have divided the analysis into 3 major sections - *User, Repository and Language*. Section 3 has a detailed explanation of this analysis.

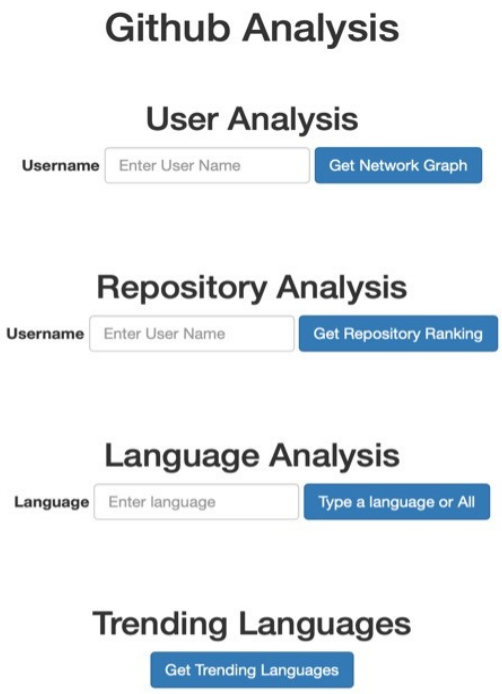


Figure 2.1: User interface

## 2.2 LIBRARIES

The following libraries were used in this project:

1. *PyGithub*: It is used to manage GitHub resources (repositories, user profiles, organizations, etc.) from Python scripts.
2. *NumPy*: It is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays.
3. *Scikit-learn*: It is a machine learning module for Python built on SciPy which is a straightforward and effective tool for data mining and data analysis.
4. *Pandas*: It is a Python package providing fast, flexible, and expressive data structures designed to make working with relational or labeled data both easy and intuitive.
5. *Bokeh*: It is a Python library for interactive visualization that targets web browsers for representation

6. *Networkx*: It is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

7. *Matplotlib*: Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc.

## 2.3 GitHub API

The API used is PyGithub which can be used to manage user profiles, repositories, issues etc. A token can be generated for every account which can be used to access all the public account in GitHub. There is a limit of 60 API calls per hour per user. We are using many different APIs to perform our analysis. Some of them are:

1. *get_user*: Takes an optional parameter as username and gets the user.

2. *get_repos*: Path to the repository have to be mentioned which will get the repository.

3. *open_issues_count*: Gets the count of all the issued that are currently open for a particular repository.

4. *stargazers_count*: Gets the total number of stars of a repository.

5. *forks_count*: Gets the total number of forks of a repository.

6. *get_followers*: Gets the count of followers for the user.

7. *get_following*: Gets the count of following for the user

8. *get_contributors*: Gets the count of all the users who have contributed for a project.

## 2.4 Data Scraping

The first two analysis - User and Repository are performed using live data. Real-time input can be provided to the application and the visualization is obtained. This is a single API call and works efficiently. Based on the user input, the result returned is not too large.

For language analysis we are trying to perform analysis on several repositories and since there is an API limit, data has to be scraped for getting the count. This is done by getting all the repositories created since 2009 till 2017. Data is scraped for every single day and details of all the public repositories created that day are obtained. For our analysis, we are scraping fields such as User Name, Repository Name, Language, Date Created. Dataset creation took about 350 hours to complete execution. We have a total of 654751 rows and 5 columns.

# 3 Analysis

## 3.1 User Analysis

Every user has a unique username. This can be used to explore some features related to a user on GitHub. Each user on GitHub can 'follow' any other user on GitHub based on their interests. They are known as 'Followers'. GitHub also keeps a track of the users you are 'Following'.

On GitHub, users don't follow another user based on friendship. They follow another user based on similar interests, project types, etc. Therefore, there will be rare cases of 2-way connections. It is possible that a user is completely stranger to another user but still follows that someone because similar interests. The more the followers of a user, it may indicate the popularity of that user based on his repositories. We are trying to analyze the popularity of a user based on this GitHub concept.

Figure 3.1 shows the network graph of a particular user. The graph has the main user in the center. The nodes are based on the followers of the main user or the users the main user is following. There are other nodes representing users who have 2-way connection with the main user indicating they are following each other. This network graph can be used to analyze the popularity of the current user among other users based on the number of followers, and the activity of the user based on the number of users the main user is following.

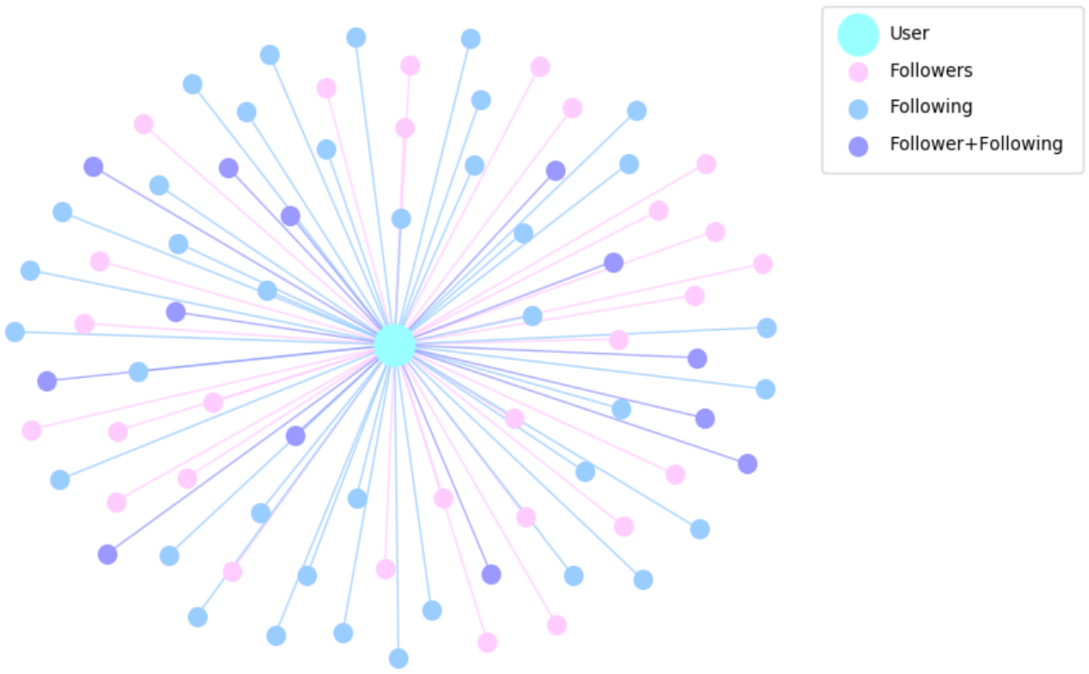


Figure 3.1: Network graph of a user and their followers

Each user has some repositories on GitHub. Repositories are important part of the GitHub as they indicate the standing of a user. They are an important criteria to rank as other users are also associated to it. Different tasks such as push, pull, watch, star, fork can be perform on any public repository. If the pushed changes are approved by the owner, the other user becomes a 'contributor' for that repository.

We are using these features of GitHub to analyze the popularity of a particular repository. The more the contributors, forks and starts for a repository, it indicates the repository is popular among other GitHub users. It might indicate that the application developed in that repository is also popular and that's the reason more users are interested in it.

Figure 3.2 shows the network graph for a particular repository of a user. The graph has a repository in the center. There are nodes representing users that are connected to the repository as contributors. There are nodes representing users who have forked the repository. There are other nodes representing users who have starred the repository. This network graph can be used to analyze the popularity of a particular repository of a user.

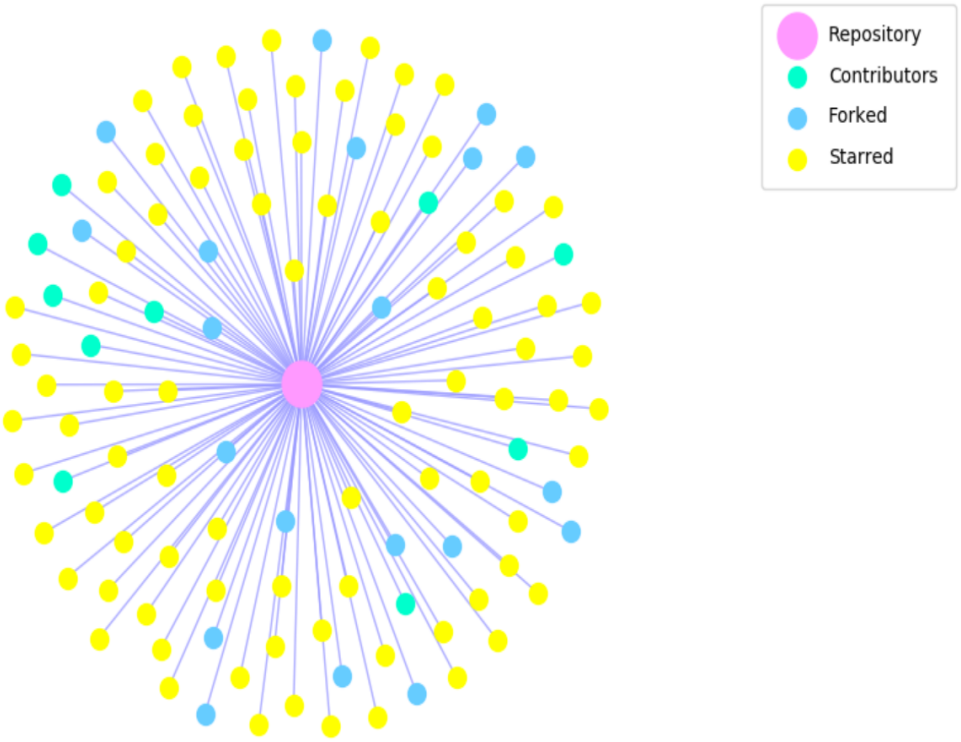


Figure 3.2: Network graph of a repository and the associated forks, stars and contributors

## 3.2 Repository Analysis

GitHub's key advantage is repository management which has several features such as fork, star, comments, issues which makes it highly social. This helps the users as they can get required information from the owner and collaborate with them to understand and enhance the projects. There are over 57 million public repositories since 2008.

Since the repositories cannot be generalized and ranked, we are ranking the repository of a particular user. This helps us recognize the good work of the user and also the improvements required for repositories that have low ranks. The ranking criteria we have used is:

$$rank = forks + stars - openissues$$

As forks are copy of a repository, it indicates that more number of users are interested in that repository. This affects the rank in a positive way. If a repository has more stars, it indicates that it contains significant content from which many users are benefited. This also affects the rank in a positive way. Instead of considering all the issues for a repository, only open issues are considered. Past issues mean that the user has been active in resolving the issues. Open issue is considered to have a negative impact in ranking

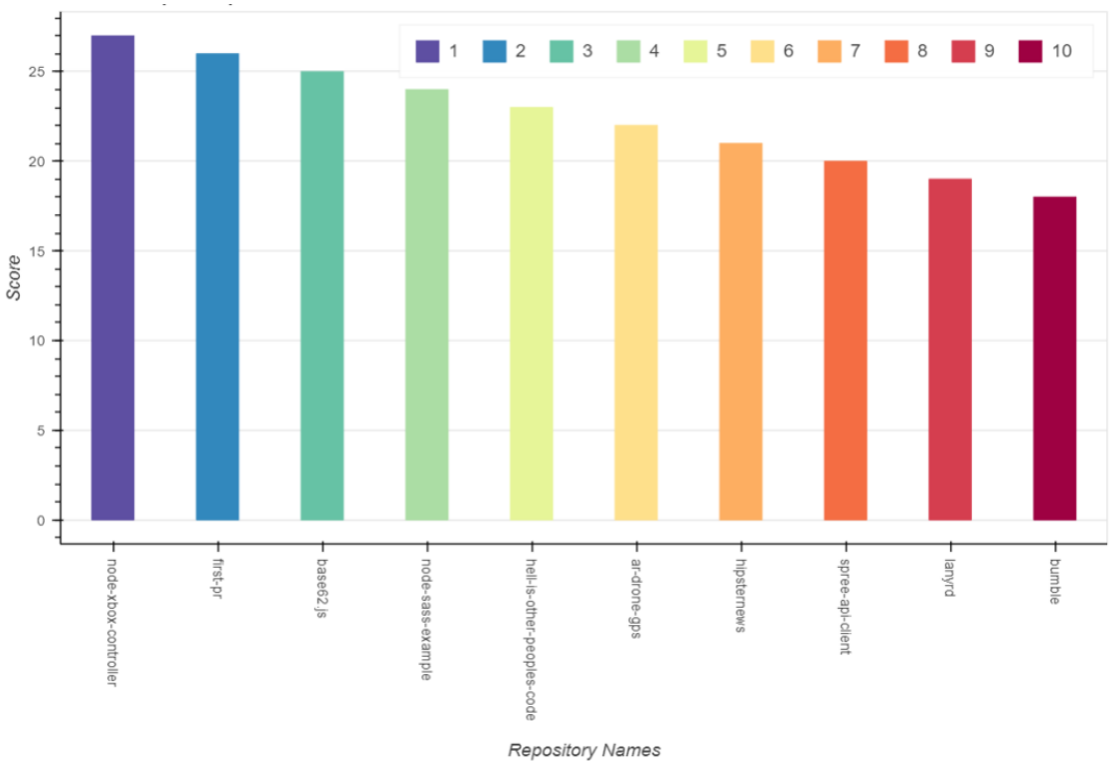


Figure 3.3: Ranks of top 10 repositories

If there are multiple repositories with the same score, same rank is assigned to all of them. Figure 3.4 shows a user with repositories having the same score. Each of them are assigned the same rank.

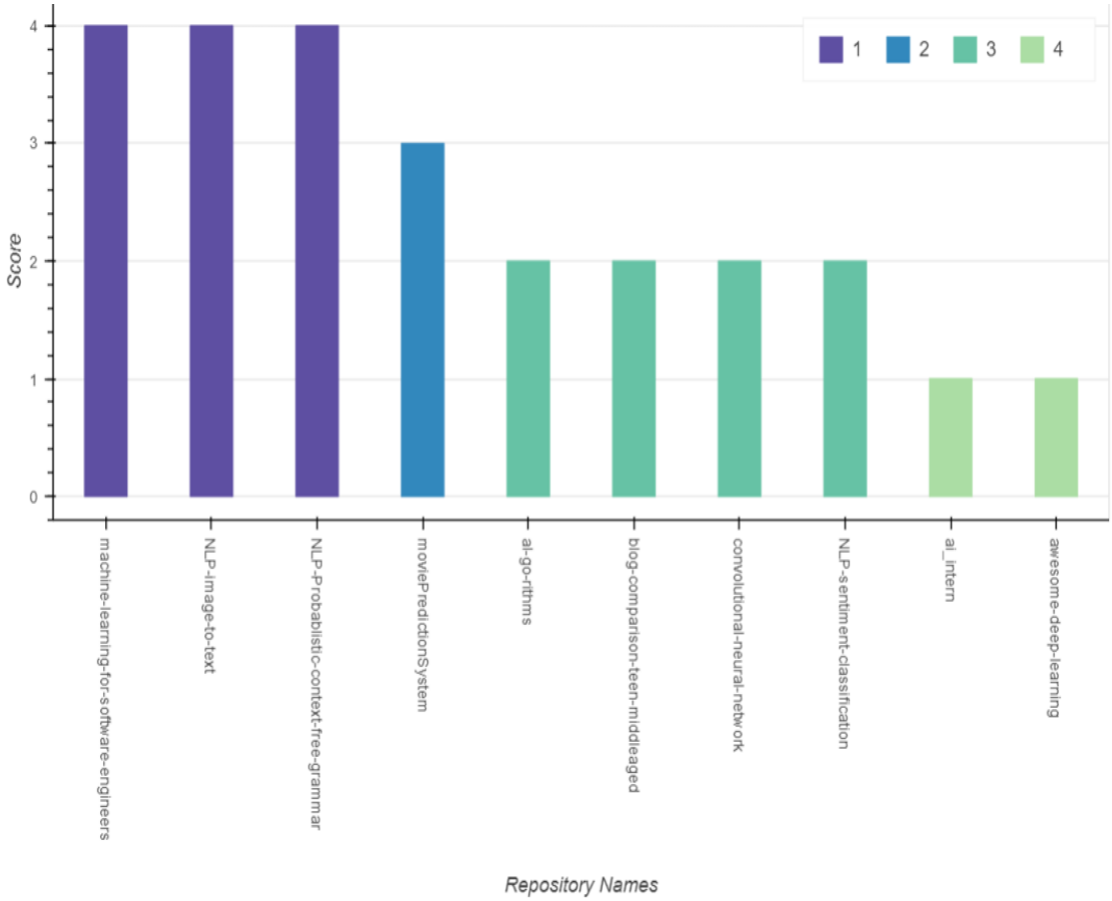


Figure 3.4: Ranks of top 10 repositories with equal score

## 3.3 Language Analysis

GitHub has repositories in over 300 languages and the number of programming languages has been increasing continuously. As a software developer, it becomes hard to keep track of all the languages and be well informed about the trending languages. The main aim of this analysis is to help developers make better decisions and stay up-to-date with the cutting-edge language.

For this analysis, we extracted 9 years data which resulted in 694660 repositories. Details of the repositories such as language used, and date created were also extracted. This was used to plot a chart which helps us identify the top languages over the past decade. Some key insights from Figure 3.5:

1. JavaScript is the most popular due to its simplicity and numerous frameworks available. Most of the major companies such as Google, Microsoft and Facebook have several implementations in JavaScript. Web applications have been developed completely using JavaScript and proves to be very powerful and efficient.

2. Java and Python being Object-Oriented languages are general-purpose languages which are highly popular among students and are used in variety of applications. Java is known for its "write once, run anywhere" philosophy which makes it popular. Python is known for being user-friendly and has numerous Data Science applications.

3. Other languages have varied applications and are used for specific purposes. Based on the requirement, these can be selected.

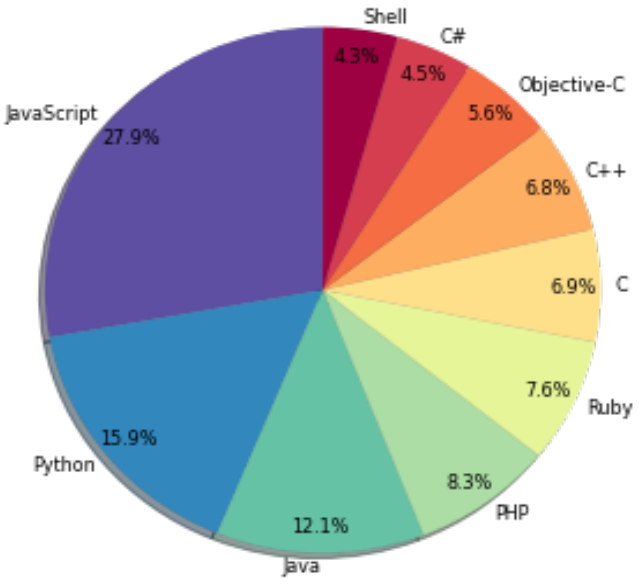


Figure 3.5: Top 10 popular languages

Trend chart helps us understand the languages which have been most used in recent times and the languages that are deprecating. Some key insights from figure Figure 3.6:

1. The count of JavaScript has been the increasing rapidly. There are about 2.3 Million JavaScript repositories as of 2018.
2. Java and Python have been constantly used due to its packages, IDE support, libraries and documentation.
3. Languages such as Ruby and C are having a significant decline in the number of repositories over the past few years.

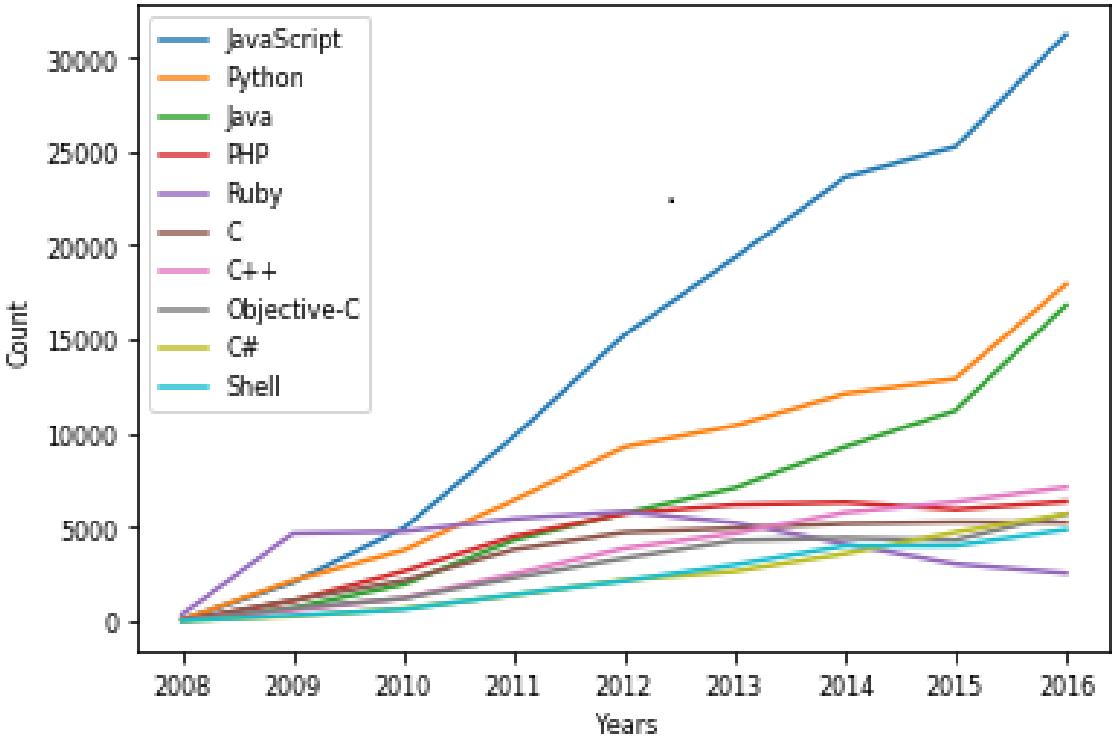


Figure 3.6: Trending languages

## 4 Conclusion

This analysis helps us better understand the users of GitHub, the languages they are using and their interests. A personalized report can be generated to each user giving details of current trends, language suggestion based on their interest and their performance status.Futher, a recommendation system can be built.

# REFERENCES

[1] https://media.readthedocs.org/pdf/pygithub/stable/pygithub.pdf

[2] https://www.oreilly.com/library/view/mining-the-social/9781449368180/ch07.html

[3] https://developer.github.com/v3/