BDP FINAL PROJECT: TURINGBOTS VS. HUMAN SOFTWARE DEVELOPERS?

Elena Gualda

03/07/2024

Agenda

- Executive Summary
- Methodology
- Data Analysis
- Conclusion
- Recommendation

Executive Summary

Can AI provide improvement in software developers' productivity and soon replace human software engineers and data scientists? The mission of this presentation is to answer just that – a deeper dive into TuringBots.

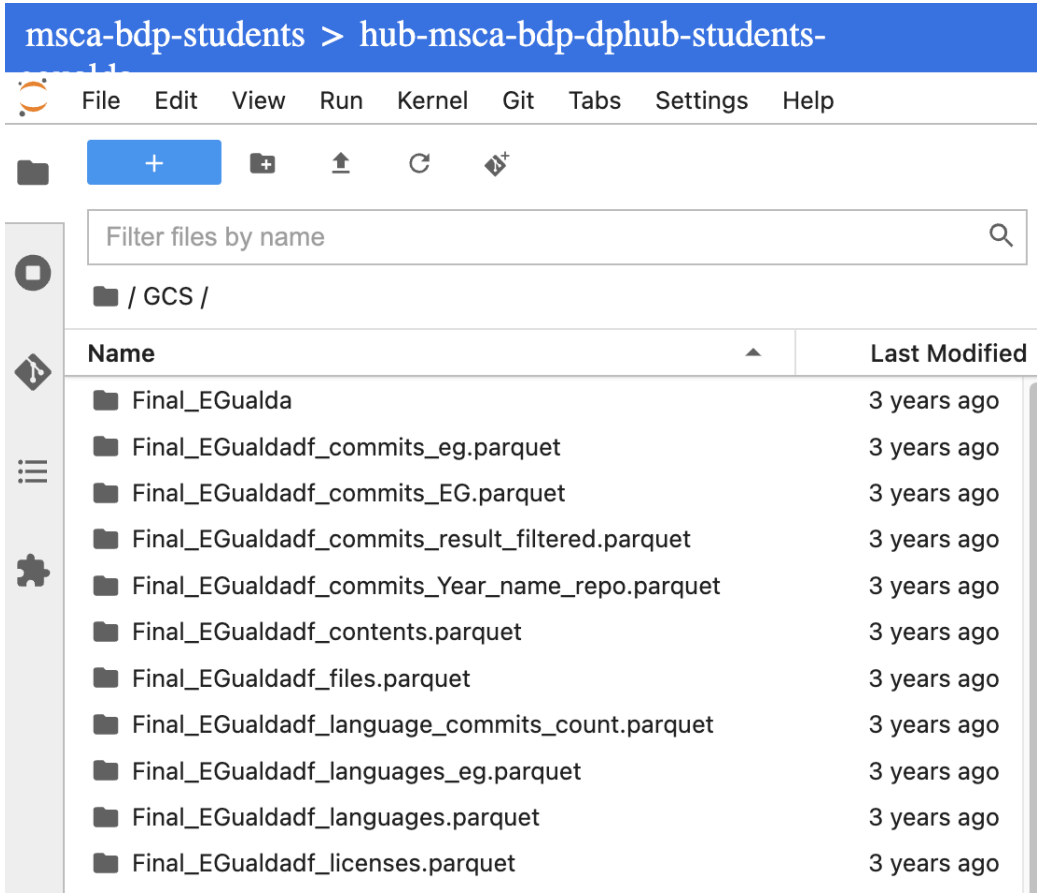
- **What we know:** TuringBots are Artificially Intelligent powered software created to aide in the development, building, testing and deployment of products or app code used and created mainly for software developers.
- **What can TuringBots do for us:** "TuringBots can automate configuration files for creating efficient DevOps pipelines; CWM TuringBots can simplify teams' collaboration and share product/project information more effectively; and development insights TuringBots can augment all team stakeholders with data insights over quality, technical debt, business value, and more." s*ourced from Forrester blogs*
- **Analysis**: With the data analysis on GitHub repositories, we will be able to take a closer look and observe the potential impact Artificial Intelligent software systems can have in the tech community.

Methodology & Data Cleanup

- Having received 5 folders with around 1.36 TeraBytes of data, parqueting each of the clean files was the best approach.
- The data received included information about GitHub repositories and commit history from 2001 – 2022. The information included metadata like author name, commit date, message, subject, language, licenses and more useful data for insight on the business question (TuringBots vs. Human Software Developers and productivity).
- In the commits file, features that were removed included 'email, trailer, and nanos' due to little importance for the data analysis. In addition, the Inter Quartile Range was calculated to gather the outlier data points to then filter them out.
- The same rules regarding feature importance were applied to the other files (Content, Languages, Files, Licenses) – variables like 'id', 'element', and 'bytes' were removed because they were of no use in the data analysis.

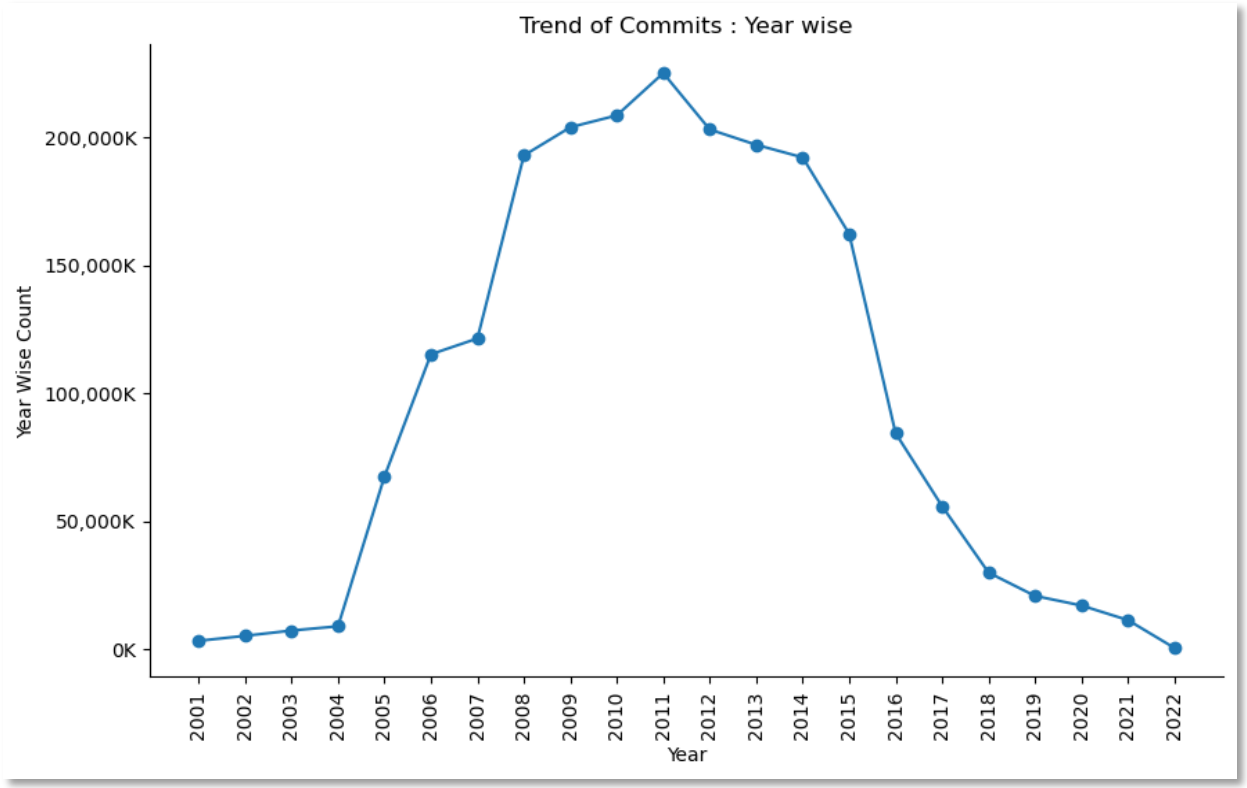
Methodology & Data Cleanup

- Due to the size of each parqueted folder, in order to efficiently clean, assess and play with the data to gather insight, I created separate notebooks for each question and had created parqueted files which contained separate aggregated data for each analysis insight.
- Having separate parquet files that were saved for my Google Cloud Cluster enabled me to easily read the needed data and join/select/sort and more with ease and confirmation that the cleaned and parsed out parqueted data would not be lost for future use!



Data Analysis:

- What is the timeline of the data? Do you see significant peaks and valleys?
  - Do you see any data collection gaps?
  - Do you see any outliers? Remove obvious outliers before plotting the timeline
  - Do you see any spikes? Are these spikes caused by real activities / events?



Insights:

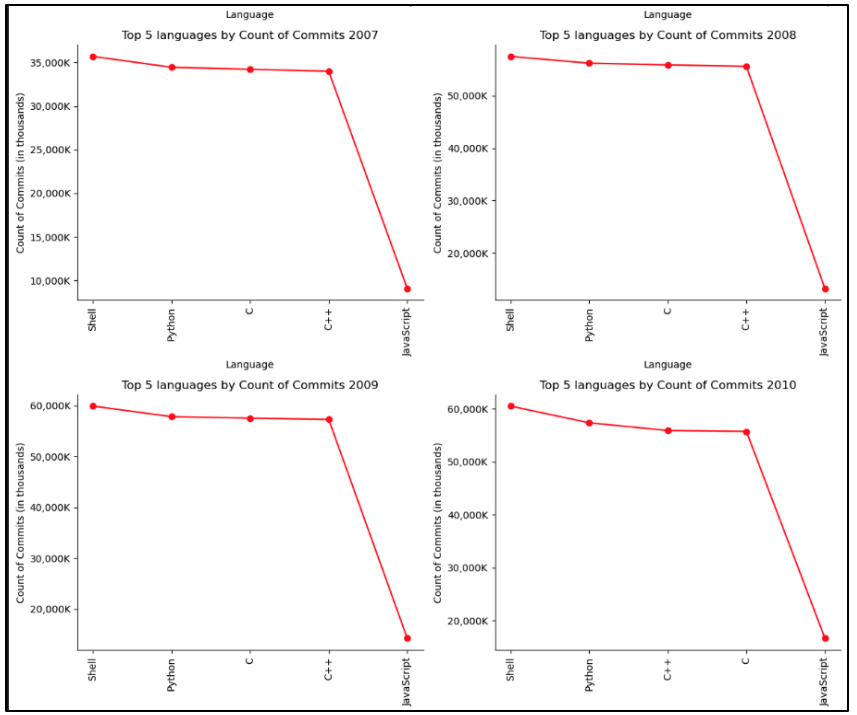
- The timeline of the data is from 2001-2022 with a high peak in 2010. After the peak there is a sharp decline in commits which could be attributed to high activity in software development especially with more open source technologies being developed and used, as well as new tech companies with OS systems like Android.
- The trends in commit activity especially the peak can also be attributed to the popularity of NoSQL rising. The evolution of the software development landscape changed for the better and with AI on the rise, it would only make sense for TuringBots to soon replace repetitive human roles.

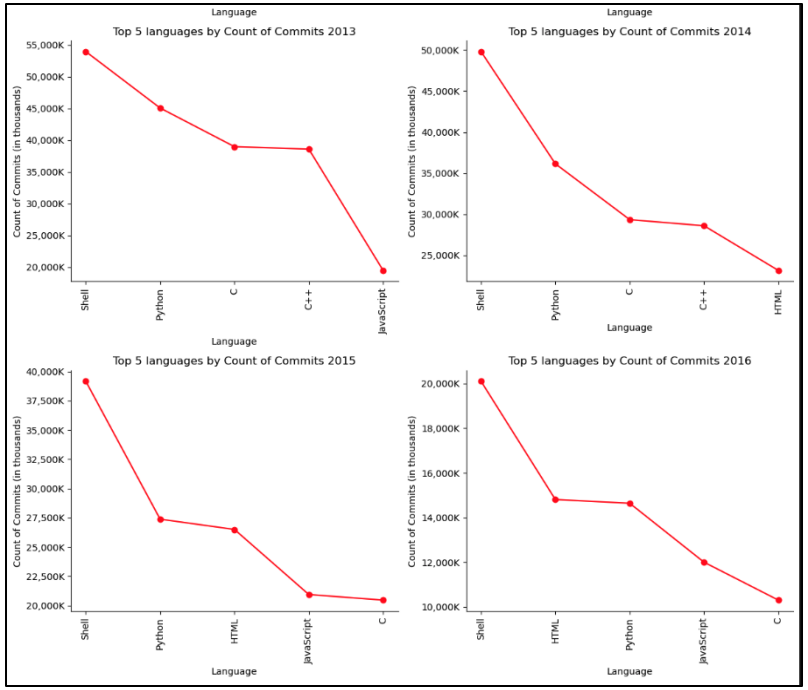
Data Analysis:

- What are the most popular programming languages on GitHub?
  - Did the trend of most popular programming languages change over time?

Insights:

- The most popular programming languages seem to always circulate around the same top 2-3 (JavaScript, Python, C). There is slight variability in other language for commits over time.
- The growth of GitHub as a platform can also contribute to the growth of the top languages.
- After the peak there is a decrease in commit counts which in turn affects the trend of languages. There could have been an evolution of more computing languages around this time as well.

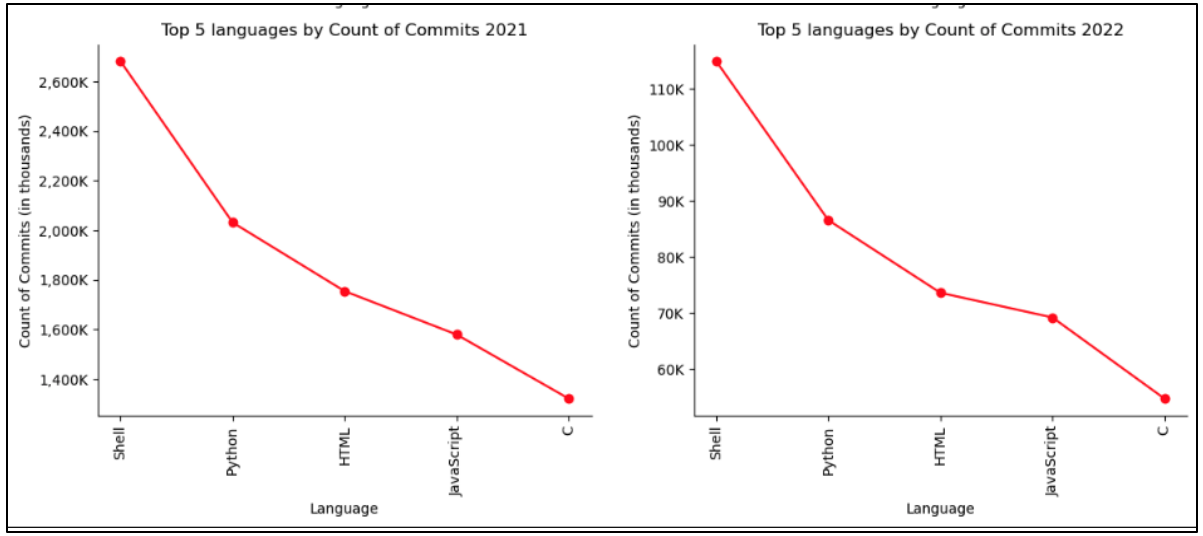


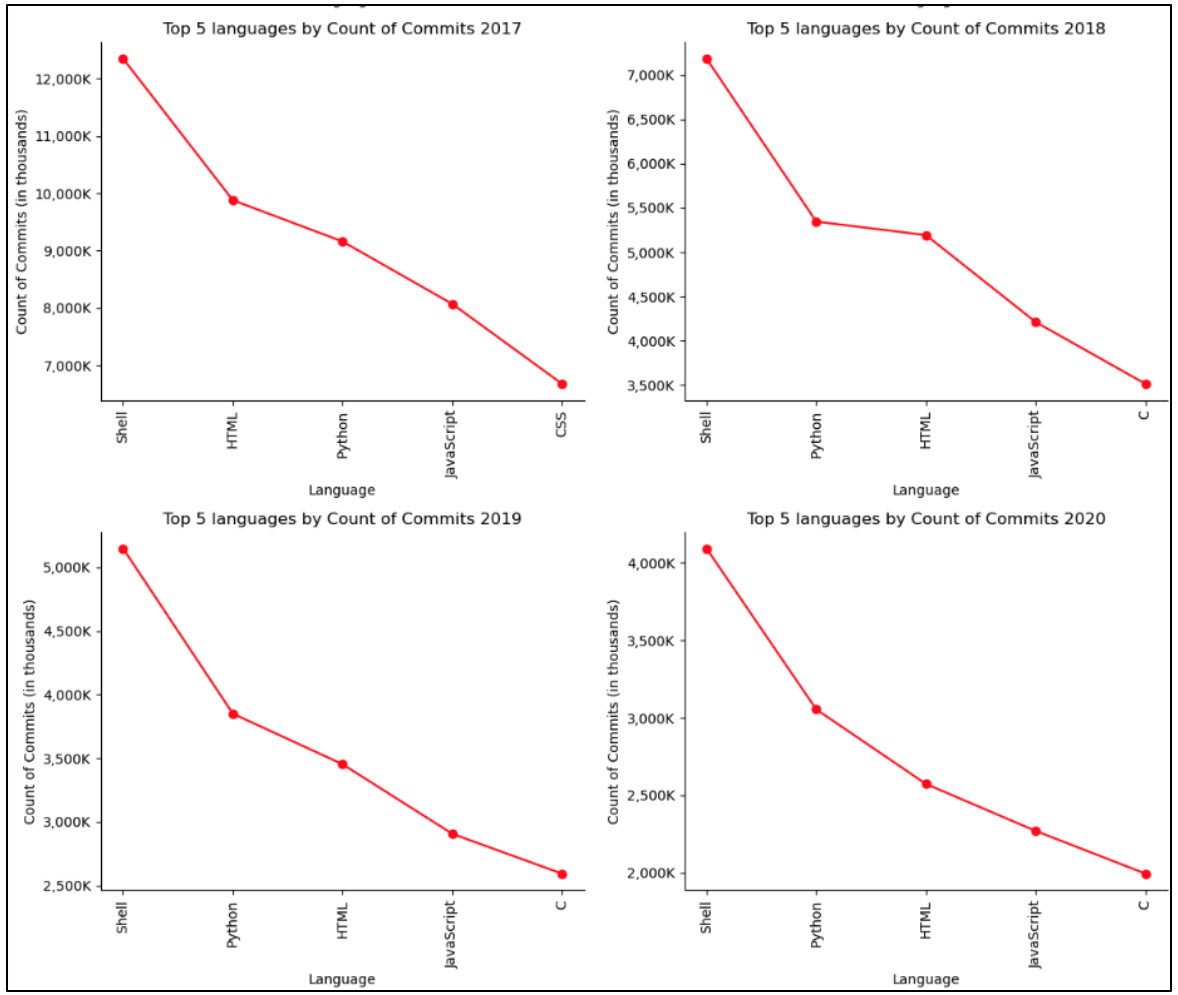


Data Analysis:

- What are the most popular programming languages on GitHub?
  - Did the trend of most popular programming languages change over time?

Continued Graphs per year



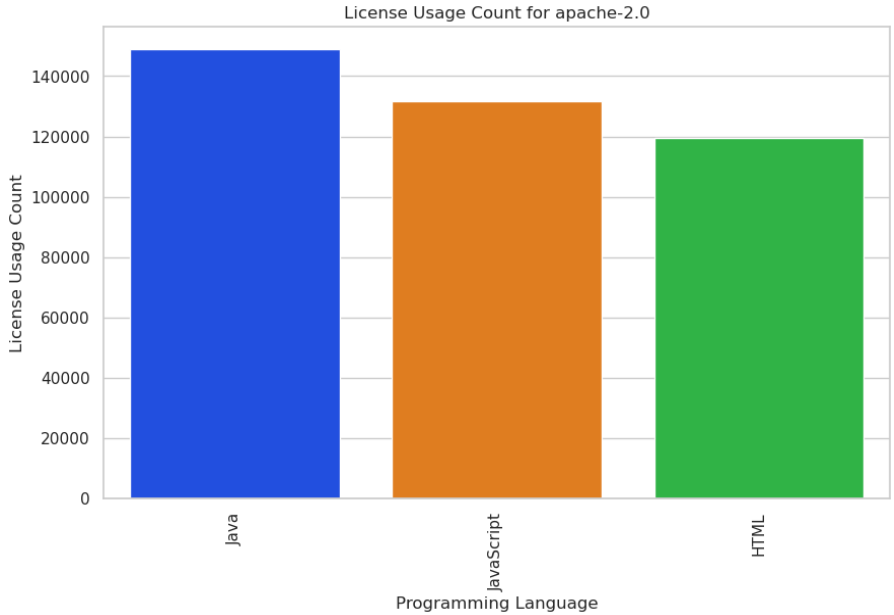


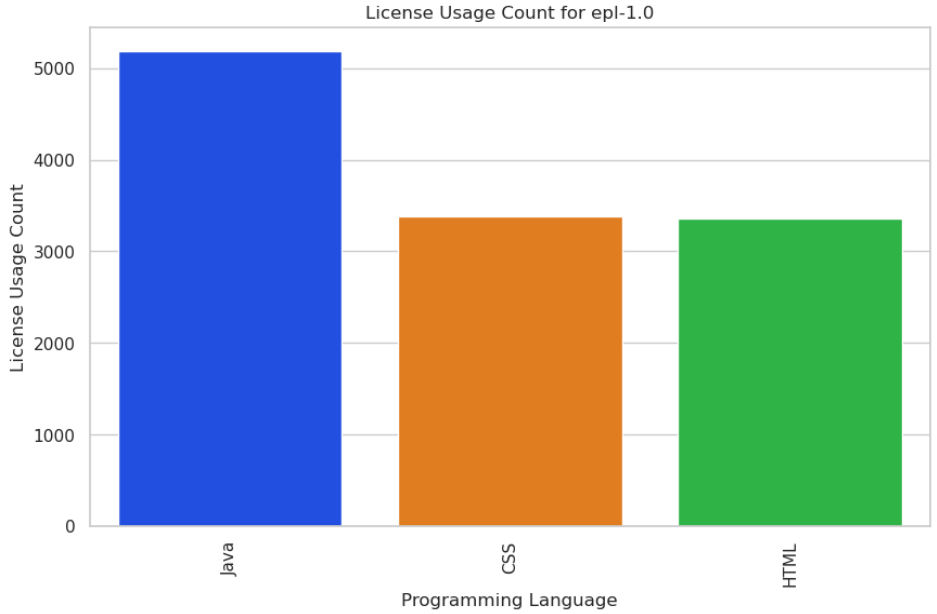
Data Analysis:

- What is the distribution of licenses across GitHub repositories?
  - Any certain programming languages that are more likely to be associated with a particular license?

Insights

- There is valuable insight that can be gained from licensure data in GitHub. After further observation, certain programming languages seem more predisposed to using specific licenses (e.g., Java and Apache – 2.0 along with EPL – 1.0 Licenses have a strong association). These observations could have occurred due to the widespread use of open-source projects that favored these licenses.
- Taking these observations into consideration when discussing TuringBots, there could be an increase in repo's that are auto-created by AI and this could lead to new and updated licensure models.
- Licensing terms will need to grow and evolve to address AI generated code, AI generated workflow systems, AI generated repositories, and AI generated/scheduled commits.



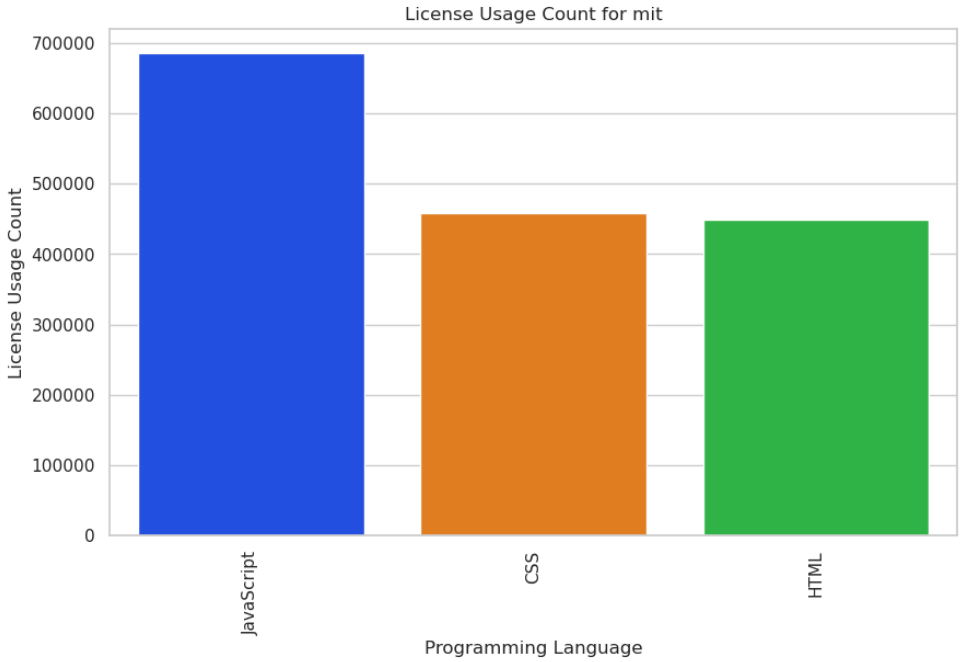


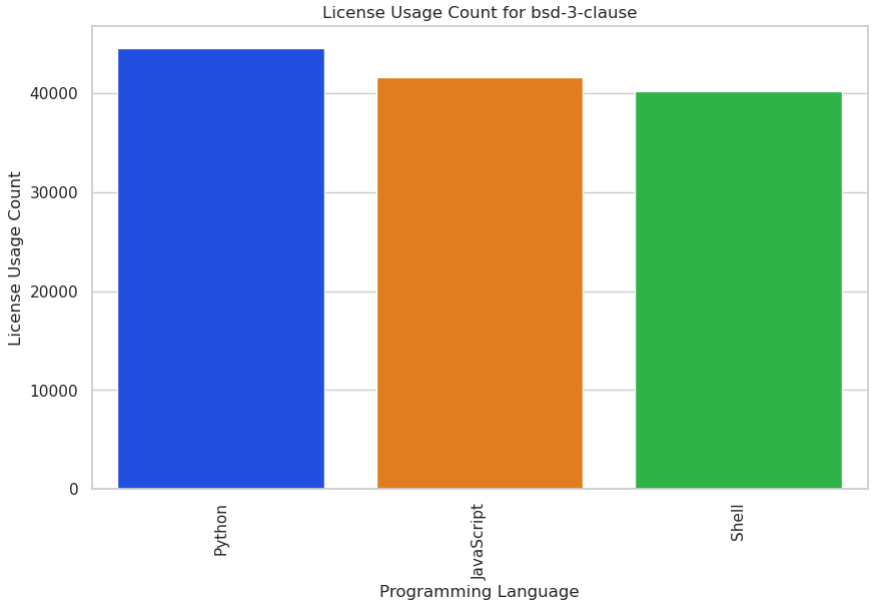
Data Analysis:

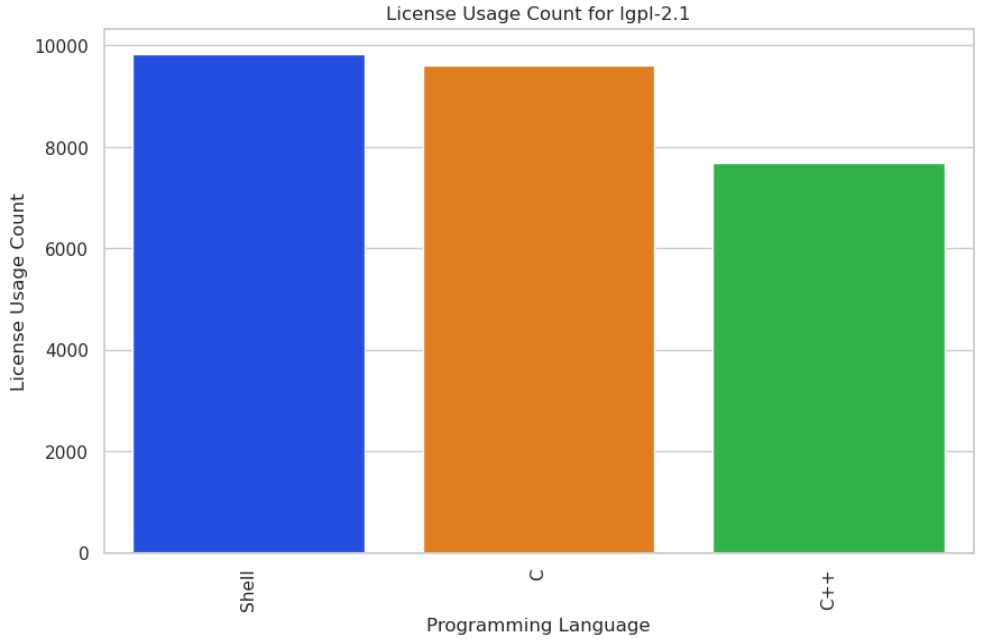
- What is the distribution of licenses across GitHub repositories?
  - Any certain programming languages that are more likely to be associated with a particular license?

Continued Graphs per year

- More examples of the sheer quantity of license usage -- upwards of 40k for Python and bsd-3 clause license/ over 600k licenses for JavaScript MIT.





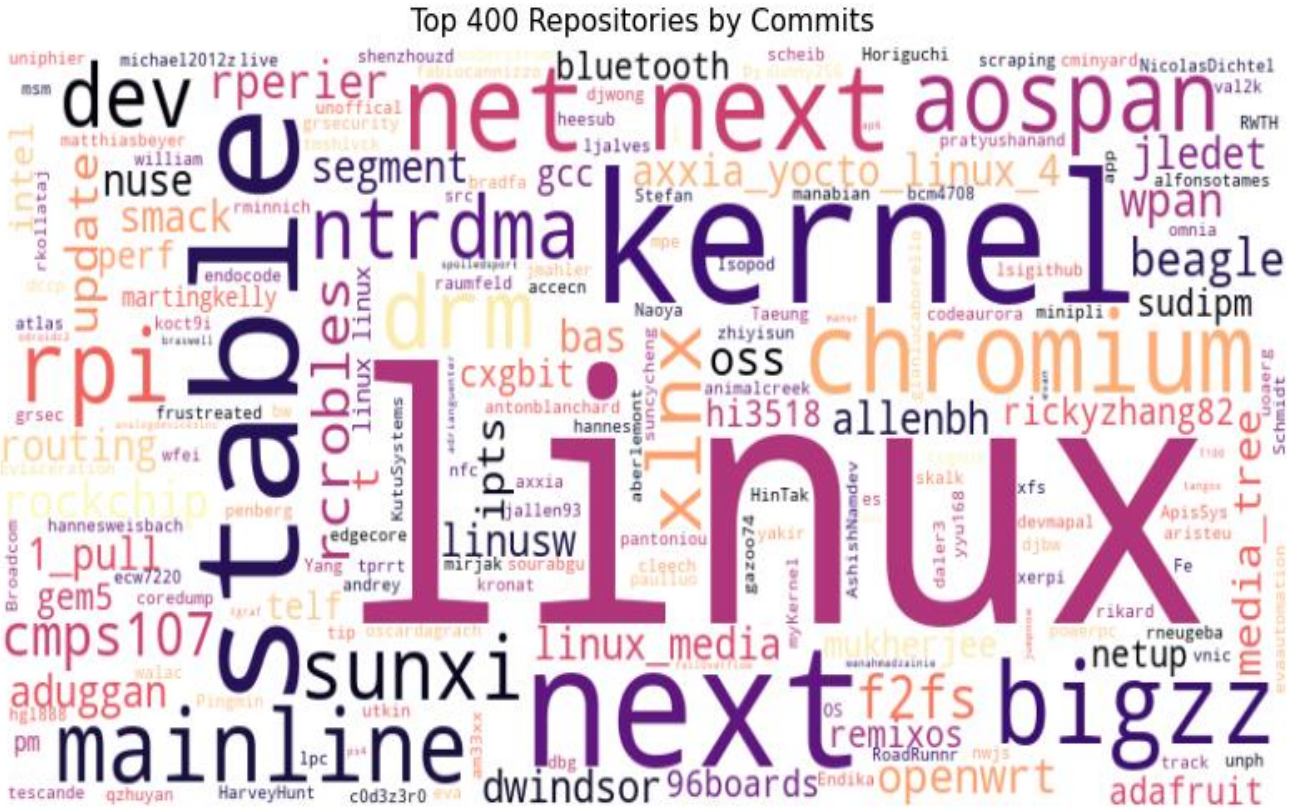


Data Analysis:

- What can you tell about the most popular and most rapidly growing repositories?
  - Is there certain technology that is driving popularity or explosive growth?
  - Are these associated with BigTech, who are open sourcing the technology?
  - Are there any technological breakthroughs that are driving this

Insights:

- Based on the wordcloud, the most popular repositories include names like "linux," "next," and "media." This could imply that areas of growth in tangent with these repositories would include operating system development, and web development frameworks.
- The technologies that are driving popularity or that have explosive growth are repos associated to Linux which usually is related to system infrastructure/web development. The same can be said for other names in the word cloud like Next for multimedia development, the list can go on!
- Examples of BigTech that might be associated or are open sourcing the technology can include IBM, they are a supporter of Linux.
- Breakthroughs in Artificial Intelligence and automation like ChatGPT/OpenAI drive these rapid growing repositories. The more that is invented and explored, the more questions asked, the more answers received and the more repositories



Data Analysis:

- Identify what technologies are most frequently associated with Data Science or AI projects
  - Did these technologies change over time?

Insights:

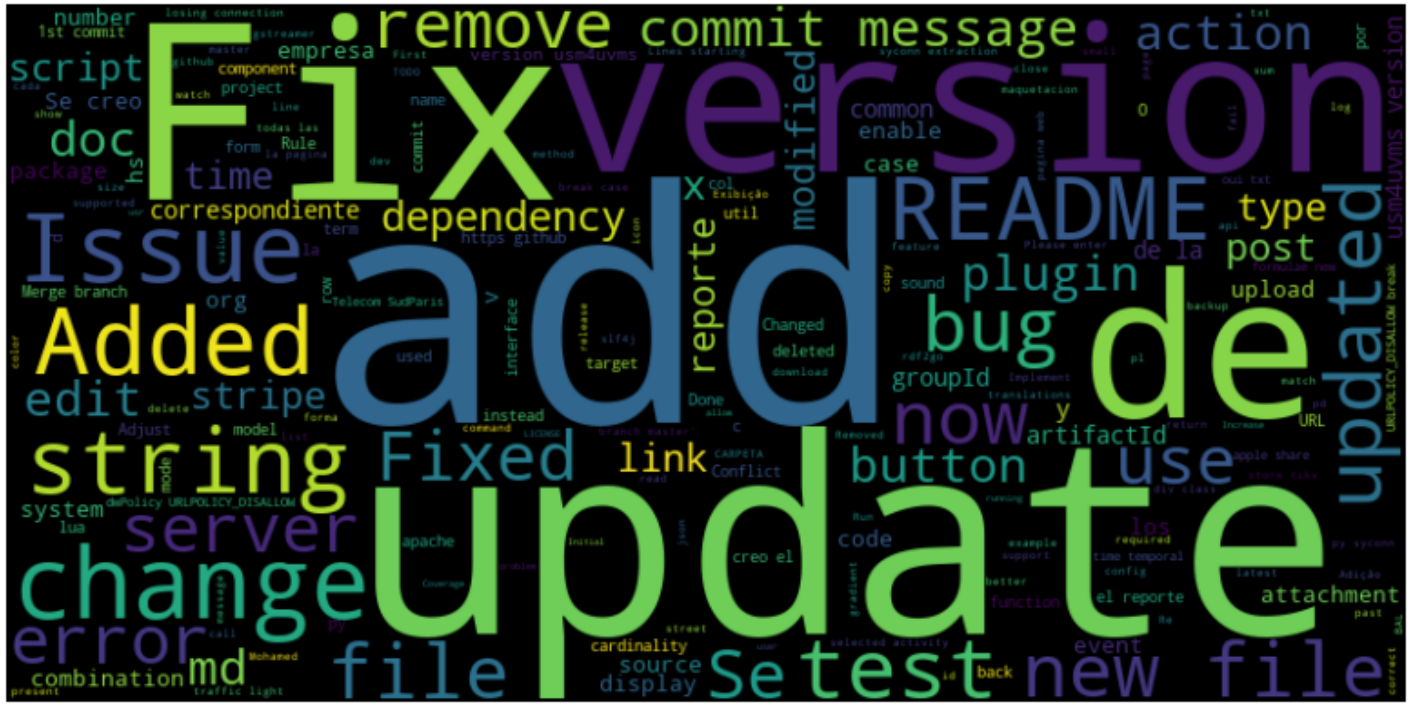
- Based on the wordcloud, the most popular technologies associated with Data Science or AI Projects include "JavaScript, Python, Perl, Chapel, C, Scilab" and more.
- Technologies like "Python" and 'Java" are some of the more prominent technologies associated with DS or AI projects and it could be due to the performance and scalability of each. In addition, with the surge of data driven analytics and machine learning, these technologies are bound to be the most frequently used and associated with such projects.
- Technologies have changed over time with the transition going from Java being the previous dominant language because of their enterprise applications to Python.

Data Analysis:

- What are the most frequent reasons for committing into GitHub repositories?
  - Is this new technology development, bug fix, etc.

Insights:

- Based on the word cloud, the most frequent reasons for committing into GitHub repositories are related to adding new code or documentation with words like "add" or "update" being present. Additionally, words like 'fix" or "bug" might show that the committer is going to commit to address a bug or error.
- Because of GitHub's ideal of continual improvement and development, these reasons for committing are most likely related to debugging or fixing projects, not so much new technology although GitHub does have the space for that.
- With TuringBots becoming more capable of completing tasks that software engineers do, there could be an increase in commits with reasons that could reflect contributions made by artificial intelligence rather than human.

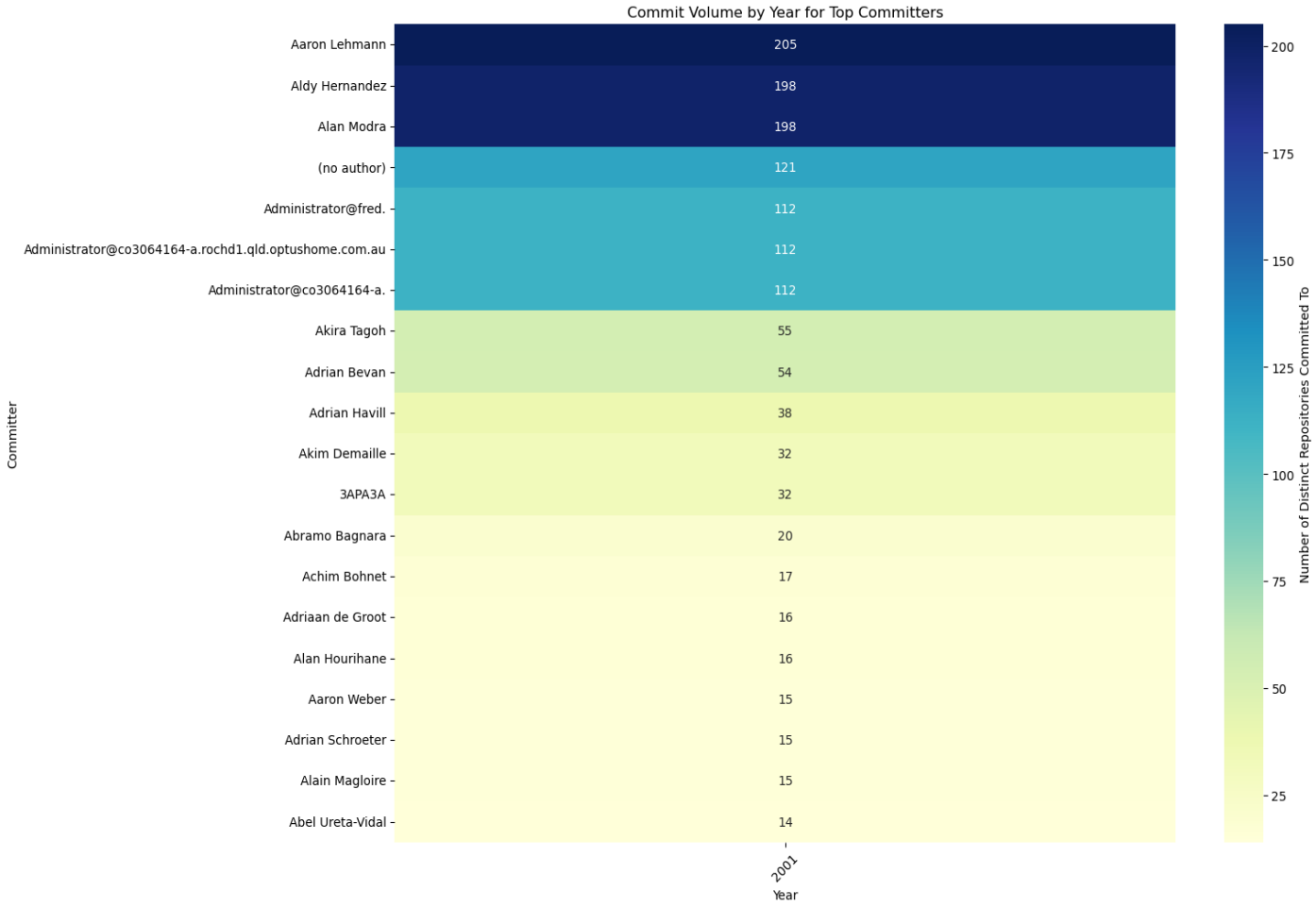


Data Analysis:

- Identify the most prolific / influential Committers
  - By commit volume
  - Visualize the distribution of these commits

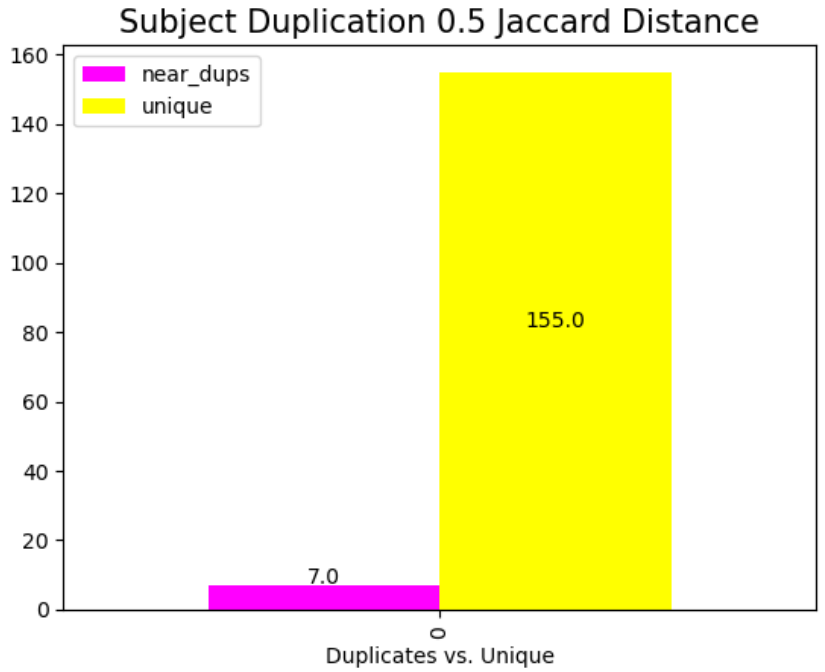
Insights:

- To the right is a heatmap of the top committers in the year 2001. GitHub was not founded until 2008 but there are commits available before it existed because the repository has been migrated through version control.
- The distribution of commits by committer for the year 2001 is skewed with a small number of individuals contributing a large volume of commits.



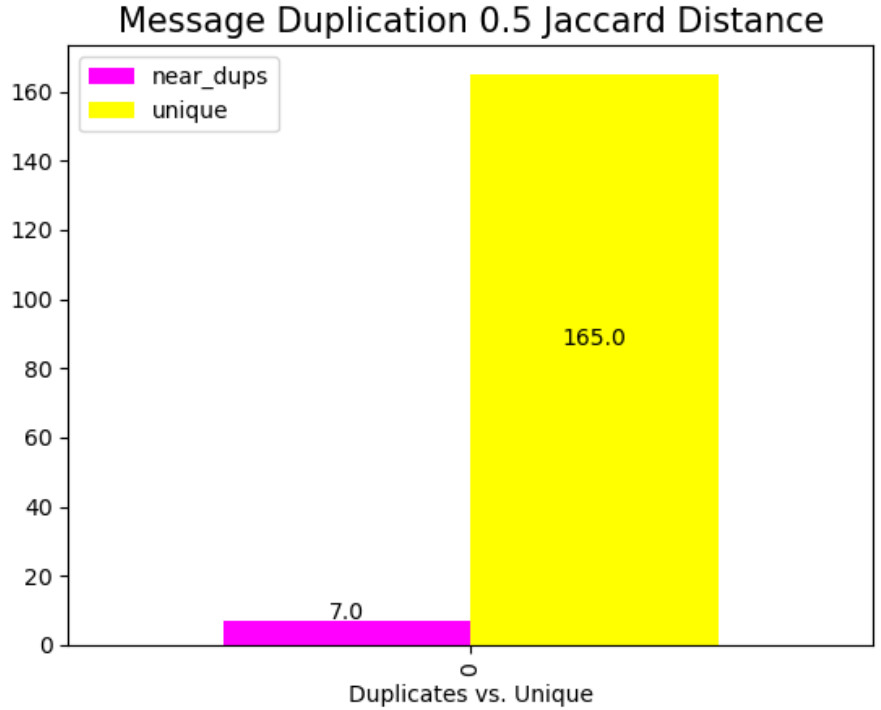
Data Analysis:

- How unique are the “subject” and “message” values?
  - Are they mostly unique? Or are people usually just copy-pasting the same text?



Insights:

- With the sheer size of the data that needed to be analyzed, the uniqueness of commit's "subject" and "message" were taken from a sample size of the data. In addition, a jaccard distance of 0.3 was too low for the similarity between the items in the dataset. Lower value jaccard means more dissimilarity so I decided to try it at 0.5 Jaccard Distance.
- Based on the visuals to the right, the jaccard distance of 0.5 for both subject and message show that there are very few near duplicate subjects or messages. This suggests that there is a high level of diversity with most subjects and messages in the sample are quite distinct from one another.
- Because of the jaccard distance of 0.5 was chosen, this signifies that the messages and subjects are quite different to be considered unique since 0.5 jaccard distance is stringent. It is evident that there is a healthy variety of topics and content within the GitHub Dataset.



Conclusion

- Based on the insight gathered from the data analysis on GitHub Commits, the potential for the use of TuringBots in replacement of software engineers is prevalent. AI and automated code generation, testing, debugging and streamlining of processes can reduce repetitive tasks and enable software developers to focus on high level solutions and software creation.
- The potential for improvement from TuringBots and artificial intelligence engines alike are inevitable and software engineers will soon have to adapt to working alongside technology as a tool.

Recommendation

- The recommendation based on the prior data analysis would be to embrace and use the software that was developed for the benefits of society. TuringBots was created to "enhance the software development lifecycle" and therefore should be used to boost productivity and enhance future pioneering.
- By adopting, creating rules and enhancing these Artificial intelligence tools, the software industry can only move forward and foster a positive environment for innovation and change.

Sources

- https://www.forrester.com/blogs/watch-out-for-turingbots-a-new-generation-of-software-development/
- https://www.cmswire.com/cms/enterprise-20/a-look-back-at-open-source-in-2010-009675.php
- https://stackoverflow.com/questions/43262563/how-far-back-does-github-track-commits
- https://integrio.net/blog/harnessing-the-power-of-turingbots-across-sdlc#:~:text=TuringBots%2C%20advanced%20AI%2Dpowered%20tools,%2C%20testing%2C%20and%20deployment%20processes.