Final Project: Git

Big Data Platforms

Vincent Caldwell December 2nd, 2023

Executive Summary

Final Project: Git

The analysis of GitHub's extensive data using AI and machine learning techniques has provided pivotal insights into the development trends within the open-source community, with crucial implications for software engineering and data science professions.

Key Findings:

Exploratory Data Analysis (EDA):
Preliminary data analysis ensured precision by converting time units, normalizing time zones, and confirming date ranges. Notably, pre-GitHub data consisting of over 3.7 billion records was identified and excluded from 2008 onwards, maintaining dataset relevance and authenticity.

Commit Activity Patterns:
The research uncovered that coding activity on GitHub tends to peak mid-week, particularly on Thursdays and Tuesdays. Based on the data's temporal distribution, distinct spikes in activity were observed during certain weeks, possibly linked to project milestones or collaborative sprints.

Programming Language Trends:
Traditional languages such as Shell, Python, C, and C++ have dominated GitHub, but trends show a shift toward web development languages like HTML and JavaScript in recent years. Additionally, the rise of newer languages, like Go, indicates a dynamic and evolving programming landscape.

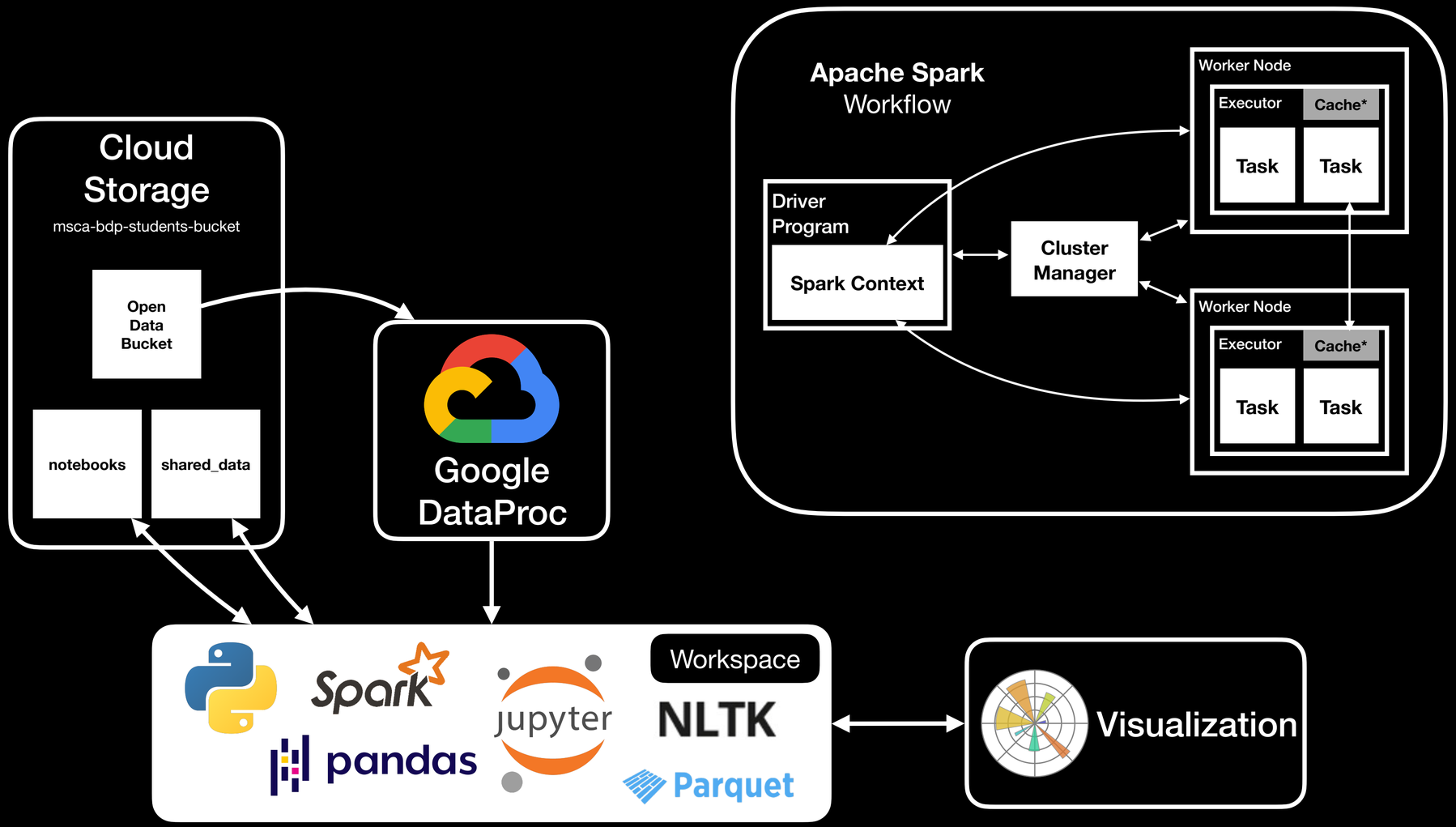
License Distribution:
GPL-2.0 emerged as the most prevalent license, signaling the strong ethos of open-source collaboration within the community. Other licenses like Apache-2.0 and MIT also maintained significant usage, offering flexibility and promoting broader collaboration efforts.

Rapid Growth Repositories:
Some of the fastest-growing repositories appeared to be test or artificially inflated projects, lacking evident contributions from technologies driving genuine growth. Key indicators suggest the presence of inflated metrics rather than technological breakthroughs underpinning the perceived rapid rise.

While AI provides powerful analytic tools, their role is to augment human capabilities not to completely replace them. The insights gleaned from such analyses can enhance productivity, guide educational focus, and inform strategic decisions within the software development and data science spheres.

THE UNIVERSITY OF CHICAGO
Vincent Caldwell December 2nd, 2023

Methodology and Source Data Overview



Data Architecture

Exploratory Data Analysis

Date Transformation:

Converted seconds to dates.
Applied a timezone offset to normalize back to the original time zone.

Date Range Confirmation:

Confirmed the date range for the dataset.

Commit Date Refinement:

Confirmed the first possible date for a commit (April 10, 2008).
Removed all dates prior to GitHub's existence.

Commit Data Frame Creation:

Created two commit data frames (prior to 2008 and after).

Prior Commit Data:
Removed dates before April 10, 2008 (GitHub start date).
Count between January 1, 1970 - April 10, 2008: 3,776,781,253 records.

After Commit Data:
Removed all dates after November 22, 2023 (day the project was started).
Count: 3,102,199,219 records.

Data Cleaning:

Removed all unnecessary columns from all tables.
No major NA values in any column used.

Data Integration:

Combined Content and Files tables.
Combined Language and Licenses tables.
Added the adjusted date from Commits.

Final PySpark DataFrames:

Commits						
commit	repo_name	author_name	committer_name	subject	message	adjusted_date

Contents_and_files						
id	repo_name	ref	path	size	content	copies

Language_w_license			
repo_name	language_type	adjusted_date	license

Definition and Significance:

Exploratory Data Analysis (EDA) is the preliminary examination of data through statistical and visual methods. It acts as a navigational compass, ensuring a nuanced understanding of the dataset for informed decision-making. EDA detects outliers, unveils patterns, and provides a diagnostic tool for data quality assurance.

Transformative Insights:

EDA transforms raw numbers into actionable insights, enabling professionals to extract narratives from the dataset. It is a systematic approach fostering robust and accurate analytical outcomes, emphasizing the importance of understanding the data landscape before delving into in-depth analyses.

THE UNIVERSITY OF CHICAGO
Vincent Caldwell December 2nd, 2023

Timeline of the Data

Commit Activity Analysis Over Time

Spike Analysis:

Day of the Week:

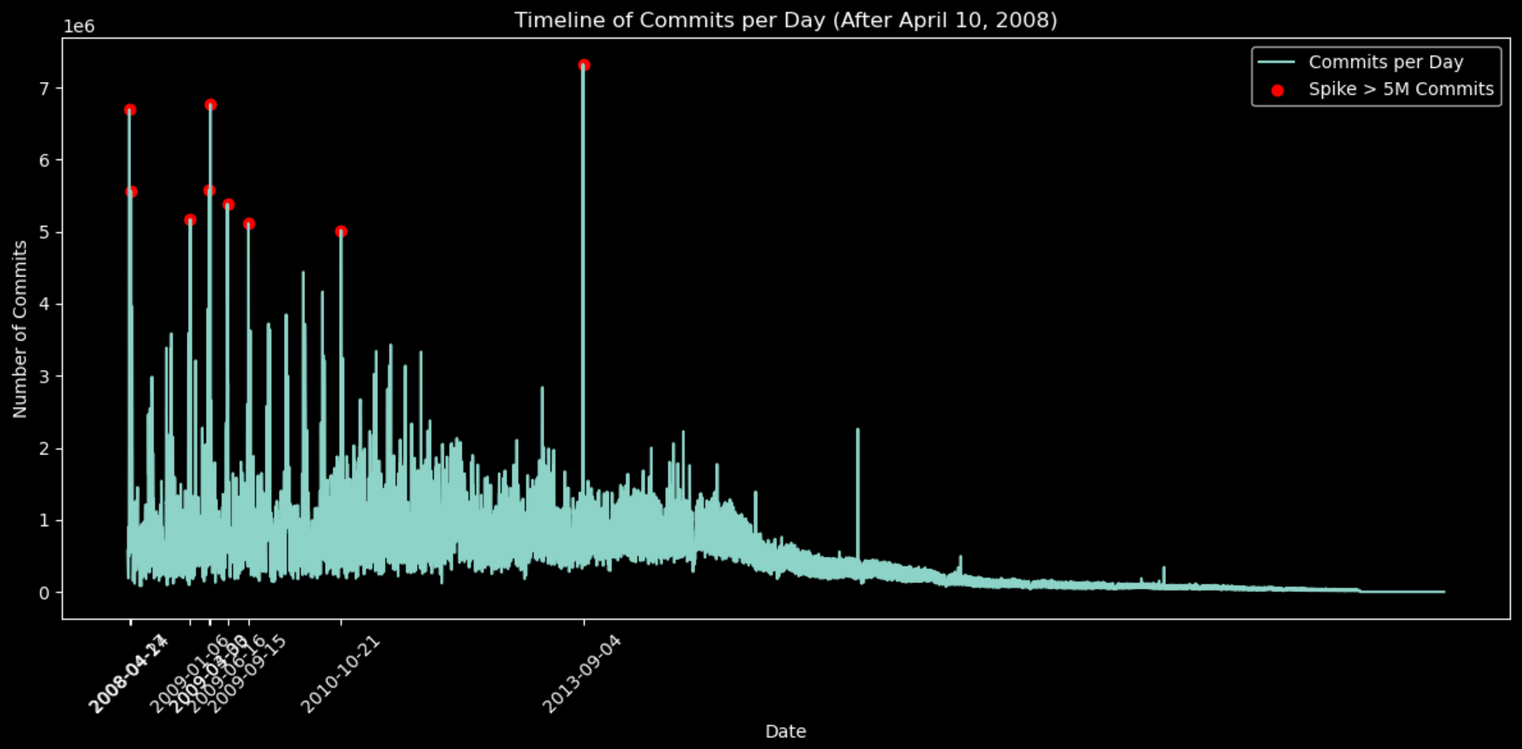
- Commit activity peaks on Thursday and Tuesday, each with a notable occurrence count of 3.
- Wednesday, Monday, and Friday also witness commits but with a less frequent occurrence count of 1 each.

Weeks of the Month:

Exploring occurrences of commits during specific weeks over various years and months provides valuable insights:

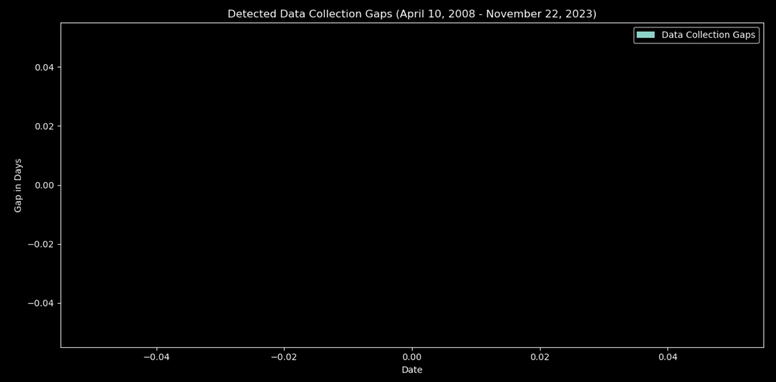
- A spike in commit activity during the fifth week of March 2009.
- Increased commits during the fourth week of April 2008.
- Another surge in commits during the third week of October 2010.
- Active development during the third week of June 2009.
- Noteworthy commits during the third week of September 2009.
- Commits during the first week of January 2009.
- Commits in the first week of September 2013.
- Commit activity during the first week of April 2009.

Our exploration of GitHub commit activity revealed significant patterns. Commits are most frequent on Thursdays and Tuesdays, each having an occurrence count of 3. Other weekdays also witness commits but at a lower frequency of 1 each. In terms of weeks in the month, notable spikes occurred during specific weeks in March 2009, April 2008, October 2010, June 2009, September 2009, January 2009, September 2013, and April 2009. These insights provide a nuanced understanding of our timeline, aiding in resource planning and optimizing development efforts.



Initial Data Collection Gaps:

Data Collection Gaps After



THE UNIVERSITY OF CHICAGO

Vincent Caldwell December 2nd, 2023

Most Popular Programming Languages on GitHub

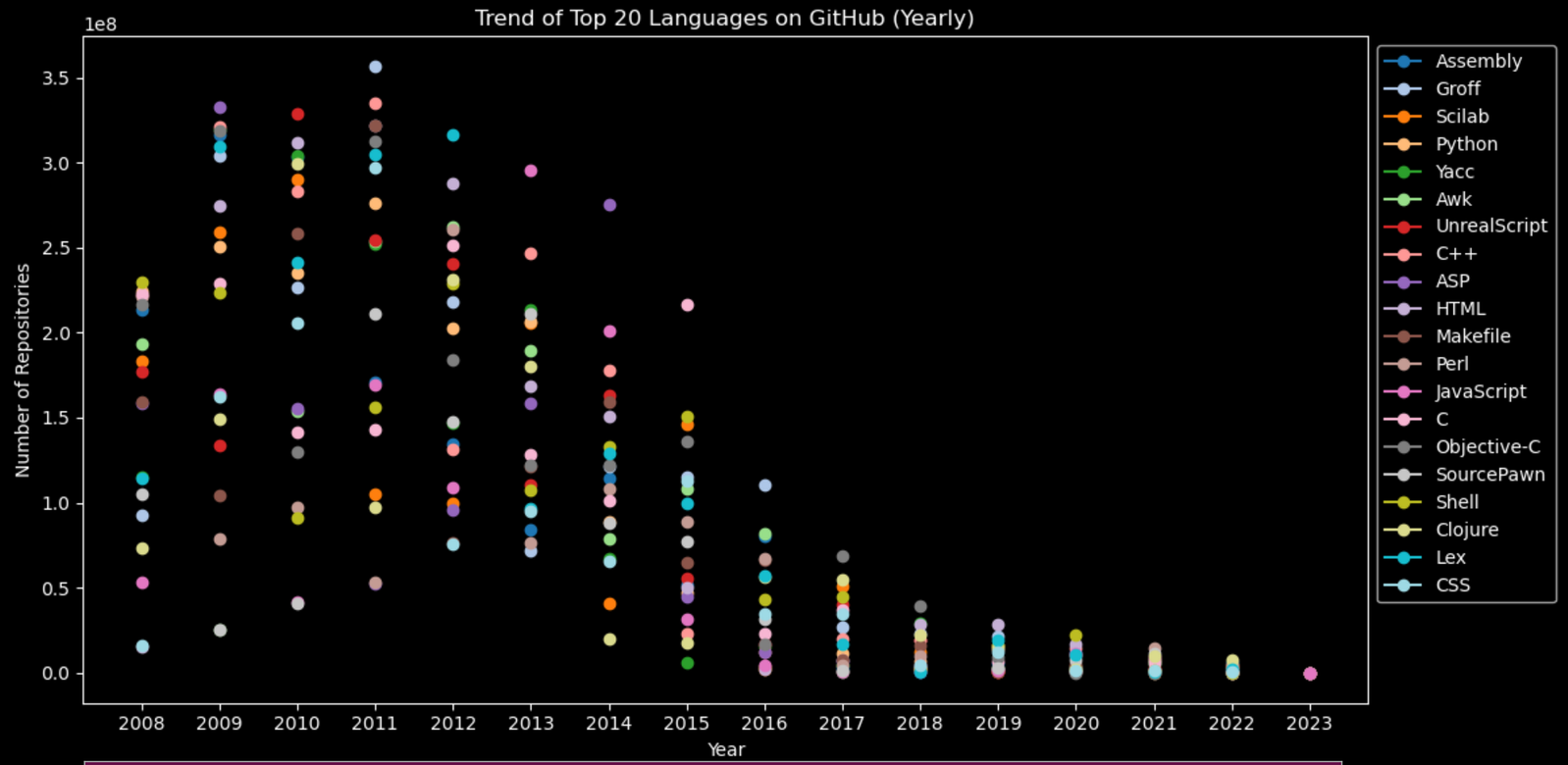
Top-5 Languages:

Shell	2,642,410,203
Python	2,293,703,436
C	2,063,789,249
C++	2,014,777,072
Perl	1,937,550,515

Language Trends Over Years:

- *2008-2013:* Consistent dominance of Shell, Python, C, C++, and Perl.
- *2014-2015:* Introduction of Makefile in the top 5.
- *2016-2019:* Rise of HTML and JavaScript, particularly in 2015-2016.
- *2020-2022:* HTML and JavaScript consistently among top languages.
- *2023:* Emerging languages like Go gaining traction.

The GitHub commit data analysis reveals a consistent dominance of languages like Shell, Python, C, C++, and Perl over the years. Noteworthy trends include the introduction of Makefile in 2014-2015 and the sustained popularity of HTML and JavaScript from 2016 onwards. The emergence of newer languages, particularly Go in 2023, indicates a dynamic shift in the programming landscape. This insight equips developers with valuable information for making informed language choices in their projects.



Popularity Change YoY (Ranked)

year	1	2	3	4	5
2008	Shell	Python	C	C++	Perl
2009	Shell	Python	C	C++	Perl
2010	Shell	Python	C++	C	Perl
2011	Shell	Python	C++	C	Perl
2012	Shell	Python	C++	C	Perl
2013	Shell	Python	C	C++	Perl
2014	Shell	Python	Makefile	C	C++
2015	Shell	Python	HTML	Makefile	JavaScript
2016	Shell	HTML	Python	Makefile	JavaScript
2017	Shell	HTML	Python	JavaScript	Makefile
2018	Shell	Python	HTML	JavaScript	Makefile
2019	Shell	Python	HTML	JavaScript	Makefile
2020	Shell	Python	HTML	JavaScript	Makefile
2021	Shell	Python	HTML	JavaScript	Makefile
2022	Shell	Python	HTML	JavaScript	Makefile
2023	C++	null	null	Go	null

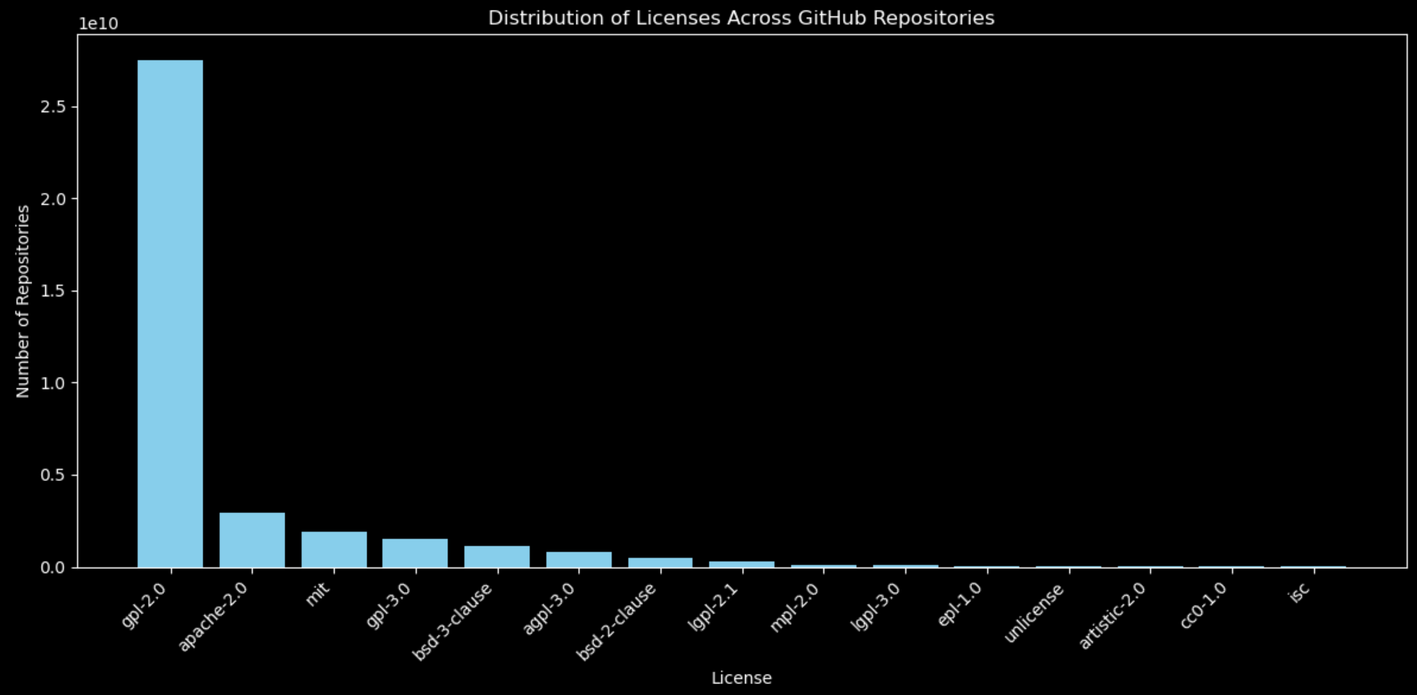
THE UNIVERSITY OF CHICAGO
Vincent Caldwell December 2nd, 2023

Distribution of Licenses Across GitHub Repositories

Top Licenses Across GitHub:

- GPL-2.0 is the most prevalent license on GitHub, with an overwhelming count of 27,506,267,723.
- Apache-2.0 and MIT licenses follow with 2,917,284,672 and 1,902,437,919 counts, respectively.
- Other prominent licenses include GPL-3.0, BSD-3-Clause, and AGPL-3.0.

Most Popular Language Per License		
Language	**License**	**Count**
Shell	**gpl-2.0**	1,763,629,754
Shell	**apache-2.0**	258,621,191
JavaScript	**mit**	215,254,737
Shell	**gpl-3.0**	110,956,860
Shell	**bsd-3-clause**	94,568,749
Ruby	**bsd-2-clause**	89,499,812
Shell	**agpl-3.0**	74,379,593
Shell	**lgpl-2.1**	21,172,361
Shell	**mpl-2.0**	13,565,644
Shell	**lgpl-3.0**	8,211,874
Shell	**cc0-1.0**	3,325,776
Java	**epl-1.0**	3,232,903
Shell	**unlicense**	3,040,643
Perl	**artistic-2.0**	1,861,301
Shell	**isc**	1,584,025



Key Insights:

- GPL-2.0 is consistently popular across languages, emphasizing its widespread usage.
- The Unlicense and CC0-1.0 licenses are notably prevalent in Shell and Python projects.
- LGPL-3.0 and MIT licenses have significant representation across multiple languages.

Recommendations:

- **GPL-2.0 Compliance:** Due to its extensive adoption, considering GPL-2.0 compliance is prudent for ensuring alignment with widely accepted open-source licensing practices.
- **Flexibility for Collaboration:** Consider permissive licenses (Unlicense, MIT) for projects aiming at broad collaboration.
- **Legal Clarity:** For languages like Java, where commercial applications are common, licenses with clear terms (EPL-1.0, Apache-2.0) may be preferable.

THE UNIVERSITY OF CHICAGO
Vincent Caldwell December 2nd, 2023

Rapidly Growing Repositories

Commit Activity Patterns for Top-20 Repos

Top Repos with High Growth:

- "comitterman/testrepo," "rankingfaker/rank...," and "dmgerman/git-test" are the top repositories with exceptionally high growth rates.
- These repositories have an average of around 2 active days but are generating a significant number of commits, indicating rapid development.

Associated Languages:

- The languages associated with high growth repositories include Shell, Python, Makefile, JavaScript, HTML, and Java.
- Shell and Python are consistently present in multiple repositories with high growth rates.

Technologies Driving Growth:

- JavaScript, Python, and HTML are among the technologies driving growth, appearing in repositories with notable growth rates.
- These technologies are commonly used in dynamic web development and scripting, suggesting a focus on web-related projects.

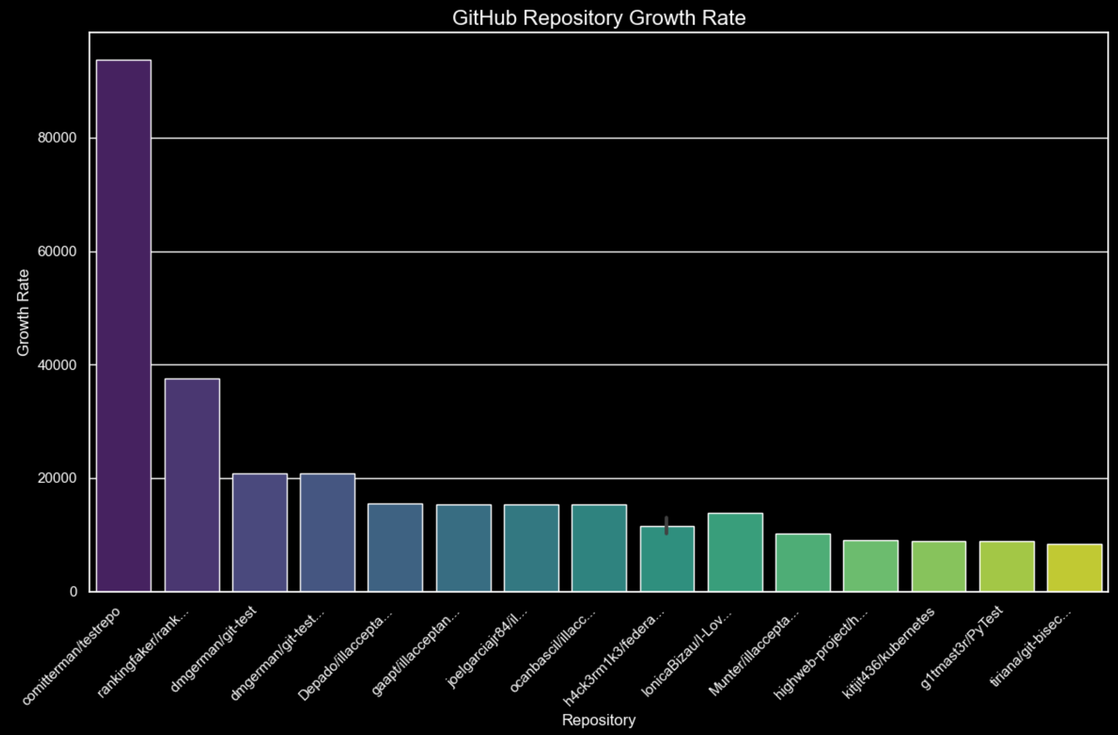
License Distribution:

- GPL-2.0 is the most prevalent license across GitHub, indicating a strong commitment to open-source collaboration and sharing.
- Apache-2.0, MIT, and GPL-3.0 are also widely adopted, reflecting diverse licensing preferences in the GitHub community.

Big Tech Influence:

- There was no representation of Big Tech[1] within the top-20 fastest growing repos on GitHub.

[1] Big Tech is defined as: Alphabet, Google, Amazon, Apple, Meta, FaceBook, Microsoft. Source: [Full list of Big Tech and Lessor Big Tech]



Trends for Rapidly Growing Repos:

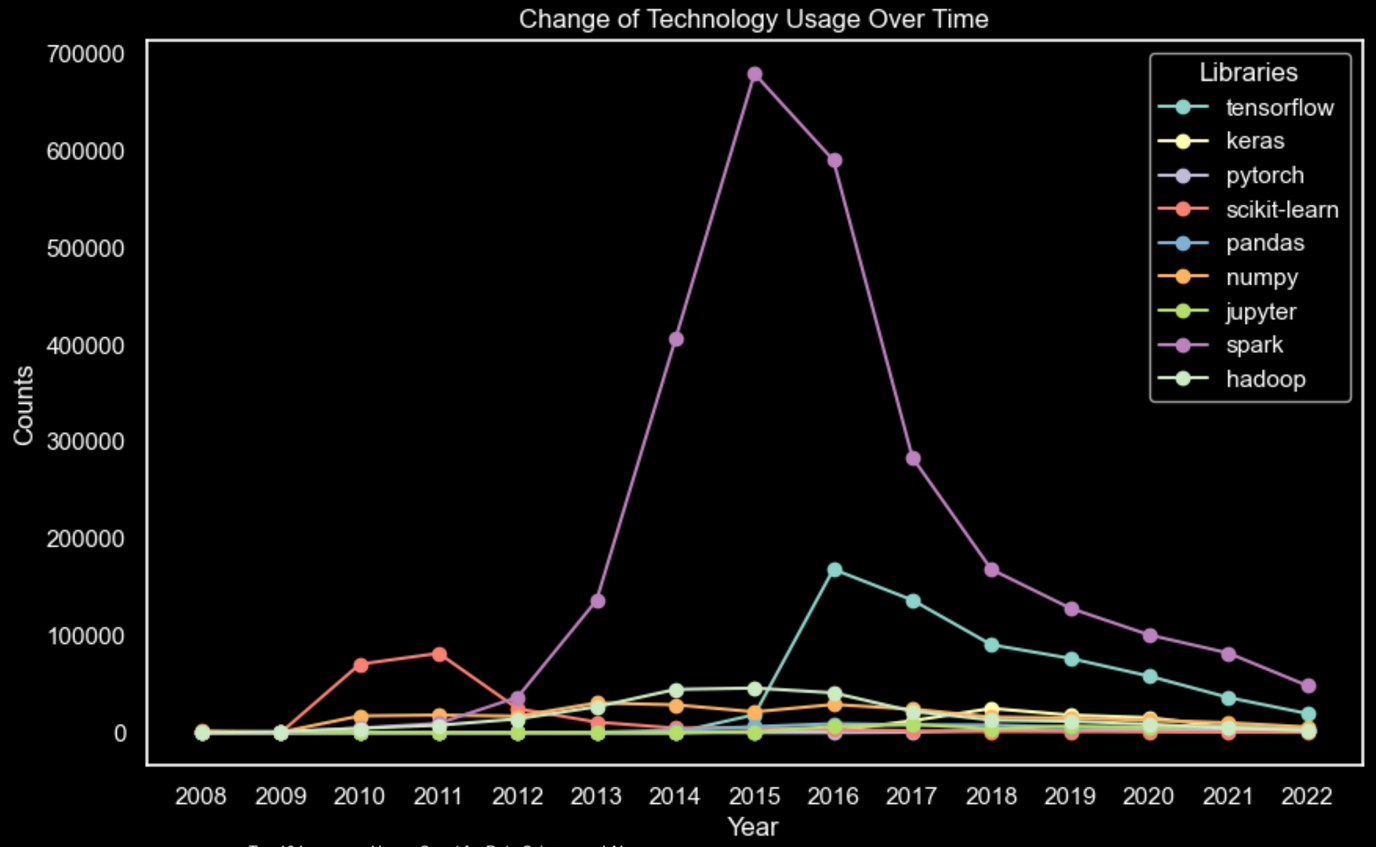
- **Breakthrough Repositories:** Examination shows the fastest-growing repositories are often test or fake designed to inflate perceived popularity. They lack signs of substantive technology contribution to their growth.
- **Encouraging Language Diversity:** Popular languages with robust community support (Python, JavaScript, HTML) should be encouraged across repositories.
- **New Technology Rising:** While there was no evidence of breakthrough technology fueling the rapid rise, technology like TensorFlow and Keras adoption aligned with Deep Learning growth.

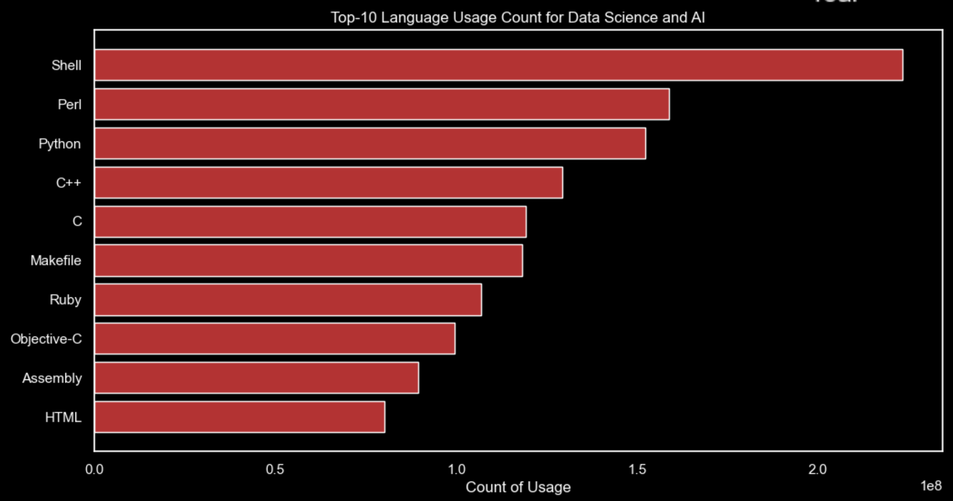
Recommendations:

To support authentic growth and technology adoption, it's crucial to foster diverse language use and open-source licensing within repositories. While analyzing high-growth repositories, we should differentiate between legitimate technology-driven growth and inflated metrics due to artificial methods. Given the rise of web technologies and Python, there is merit in concentrating on frameworks that facilitate web development. Concurrently, prioritizing ongoing education in emerging tech areas, supporting community engagement, and ensuring robust security practices are essential for maintaining a forward-thinking, secure development environment.

THE UNIVERSITY OF CHICAGO

Vincent Caldwell December 2nd, 2023





Technology	Times Used
Scikit-Learn	7,274,363
Tensorflow	6,549,117
Spark	3,793,902
Keras	26,965
Hadoop	12,652
Jupyter	3,963
PyTorch	2,832
NumPy	912
Pandas	0

Data Science & AI Technologies: Adoption Trends

Primary Language:

Python stands out as the predominant language for DS/AI projects.

Key Frameworks:

TensorFlow and Keras have seen significant and rapid growth aligned with the rise of Deep Learning.

Steady Utilization:

Scikit-learn, Spark, and Hadoop have stable usage, indicating continued relevance in Data Processing.

Trends Over Time:

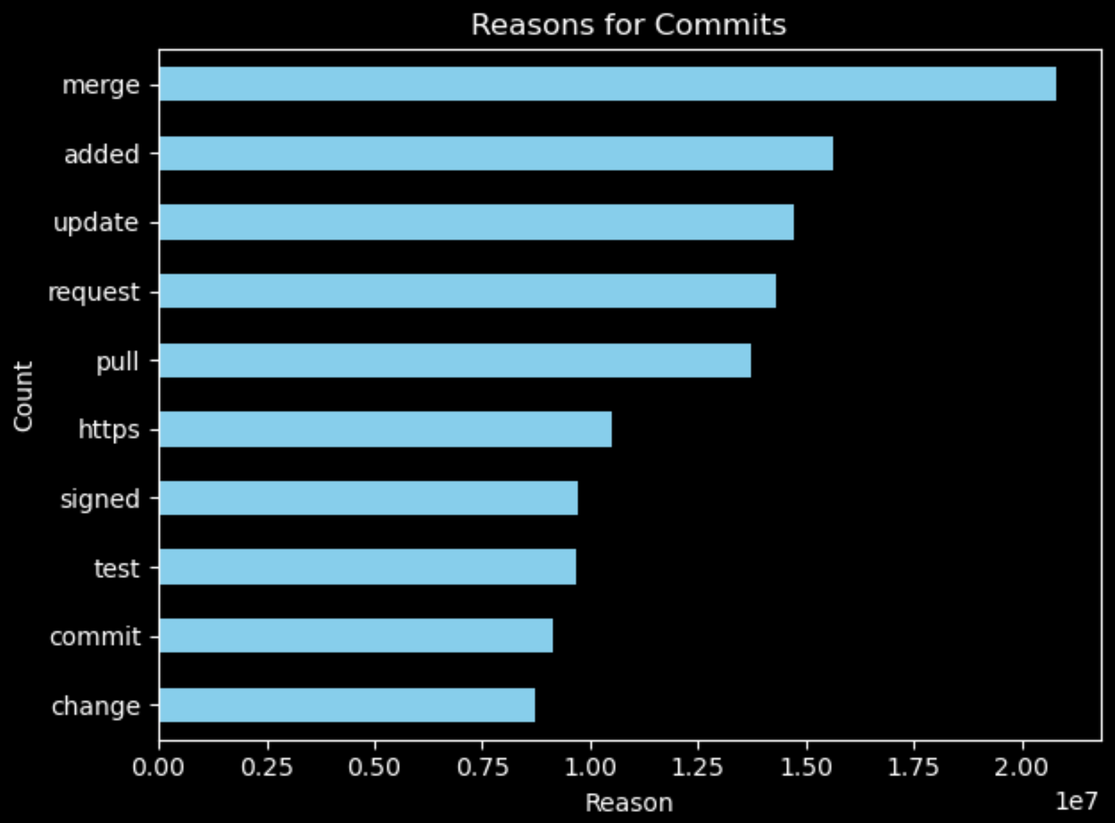
- TensorFlow's popularity skyrocketed, particularly mid-2010s onwards.
- Scikit-learn, while peaking mid-2010s, shows a decline as the community may be shifting focus to more complex tools.
- Spark and Hadoop reflect ongoing importance in handling Big Data challenges.

Insights for Strategy:

- Training, tool development, and resource allocation should prioritize technologies with growing adoption.
- Performance analysis should factor in how technology trends evolve to stay ahead in the DS/AI industry.

THE UNIVERSITY OF CHICAGO
Vincent Caldwell December 2nd, 2023

Most Frequent Reasons for Committing into GitHub



Latent Dirichlet Allocation Topic Modeling:

Latent Dirichlet Allocation (LDA) is a probabilistic model used in natural language processing and machine learning for topic modeling. It assumes that documents are mixtures of topics, and each topic is a mixture of words. LDA helps uncover hidden thematic structures within a collection of texts.

We used this to find the most frequent reasons for committing into GitHub repositories vary based on the nature of the projects and their respective needs. Analyzing the provided data, common themes emerge:

Bug Fixes and Code Modifications (Topic 1):

- Seen in terms like "fixed," "added," and "error," indicating active efforts to enhance code quality and resolve issues.

Code Structure Refinement and Testing (Topic 2):

- Highlighted by terms such as "code," "remove," and "move," suggesting continuous improvement in code organization and the management of test suites.

Kernel and File System Updates (Topic 3):

- Reflected in terms like "signed," "gmail," and "linux," indicating significant contributions related to kernel and file system enhancements.

Repository Configuration and Issue Tracking (Topic 4):

- Evident from terms like "commit," "removed," and "config," suggesting activities related to repository configuration, version control, and issue tracking.

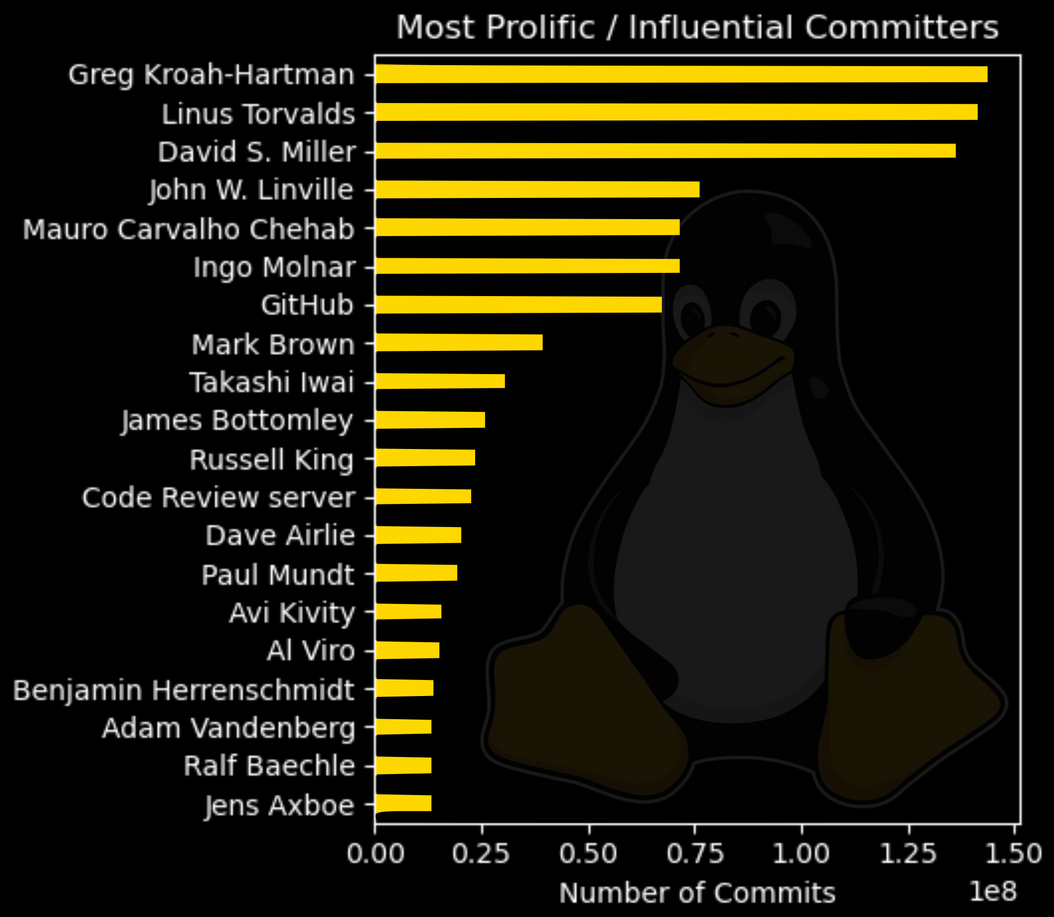
Branch Management and Feature Integration (Topic 5):

- Highlighted by terms like "merge," "request," and "pull," indicating a focus on branch management and the integration of new features

Latent Dirichlet Allocation is a powerful tool for extracting meaningful topics from large sets of text data. Its applications range from content organization and recommendation to understanding thematic structures, making it a valuable asset in uncovering hidden patterns and insights within textual information.

THE UNIVERSITY OF CHICAGO
Vincent Caldwell December 2nd, 2023

Prolific and Influential Committers



Greg Kroah-Hartman (144,043,929 commits):
- Known for: Significant contributions to the Linux kernel, maintaining the stable branch.

Linus Torvalds (141,857,676 commits):
- Known for: Creating the Linux kernel, revolutionizing open-source development.

David S. Miller (136,834,108 commits):
- Known for: Extensive work on networking in the Linux kernel.

John W. Linville (76,511,641 commits):
- Known for: Linux wireless subsystem maintenance.

Mauro Carvalho Chehab (72,218,949 commits):
- Known for: Multimedia-related contributions in the Linux kernel.

Ingo Molnar (71,870,895 commits):
- Known for: Performance-related enhancements in the Linux kernel.

GitHub (67,980,913 commits):
- Known for: Hosting a diverse array of open-source projects and fostering collaboration.

Mark Brown (40,003,222 commits):
- Known for: Audio subsystem maintenance in the Linux kernel.

Takashi Iwai (31,033,980 commits):
- Known for: Audio-related contributions and maintaining sound subsystems.

James Bottomley (26,249,448 commits):
- Known for: Work on various Linux subsystems and promoting open-source initiatives.

GitHub's key contributors, such as Linus Torvalds and Greg Kroah-Hartman, play a pivotal role in shaping critical projects like the Linux kernel. Their significant influence not only drives innovation but also establishes GitHub as a vital hub for collaborative development. By recognizing their expertise, GitHub users gain insights into best practices and cutting-edge advancements, enhancing the platform's value as a global resource for developers.

THE UNIVERSITY OF CHICAGO
Vincent Caldwell December 2nd, 2023

Text Similarity Analysis*

Text Similarity Overview:

Measures how closely two pieces of text are related in terms of content and context.
Utilized in applications such as search engines, plagiarism detection, and document clustering.

Locality Sensitive Hashing (LSH):

Algorithm used to group similar items into "buckets" using hashing.
Enhances the efficiency of finding approximate nearest neighbors in large datasets.
Particularly useful when dealing with high-dimensional data like text.

Jaccard Distance:

A statistic used for gauging the dissimilarity and diversity between sample sets.
The formula is 1 minus the size of the intersection divided by the size of the union of the sample sets.
Lower Jaccard Distance means higher similarity.

Workflow in PySpark:

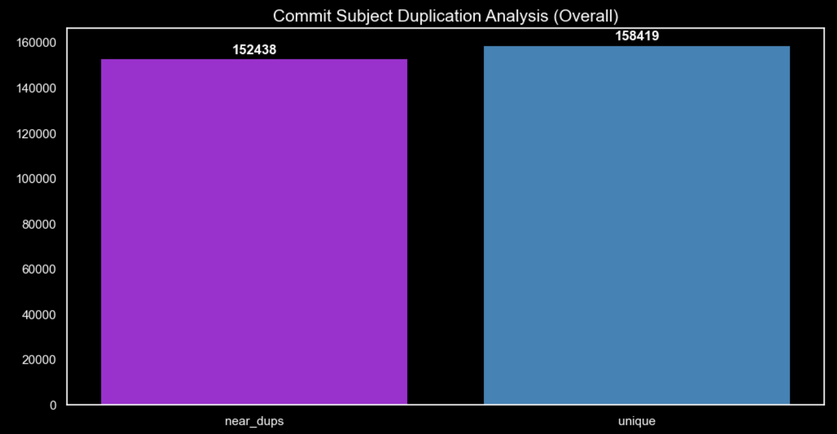
Convert text into vectors.
Apply LSH to create hash tables that map similar text vectors to the same buckets.
Use Jaccard Distance to quantify similarity between items within the same bucket.

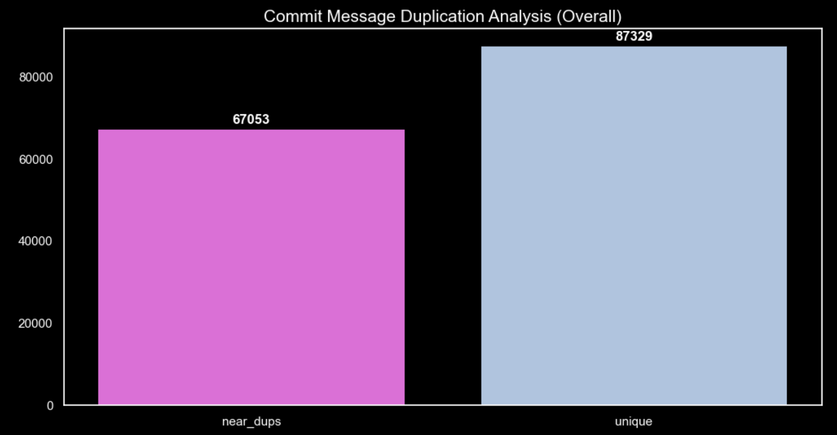
Benefits in PySpark:

Leverages Spark's distributed computing power to process large text corpora efficiently.
Provides scalable solutions for real-time applications in text analysis.

Incorporating LSH and Jaccard Distance in PySpark enables efficient and scalable computation of text similarity, forming the backbone of many modern text analysis tools.

“Subject” and “Message” Duplication Across All Commits





Conclusion: Based on a Jaccard distance threshold of 0.3, roughly half of the "subject" records and slightly less than half of the "message" records are classified as duplicates, suggesting a moderate level of uniqueness within this dataset.

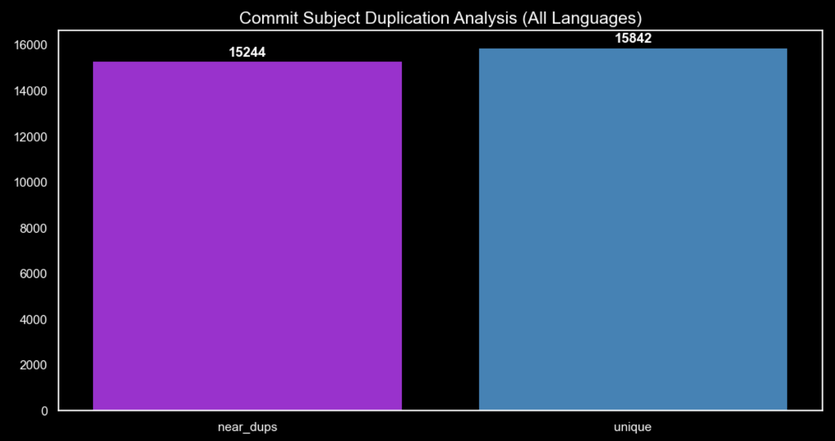
This implies duplication or commonality in topics and content being discussed in subjects and messages, highlighting areas for potential consolidation or emphasis on promoting more diverse discussions in project communications.

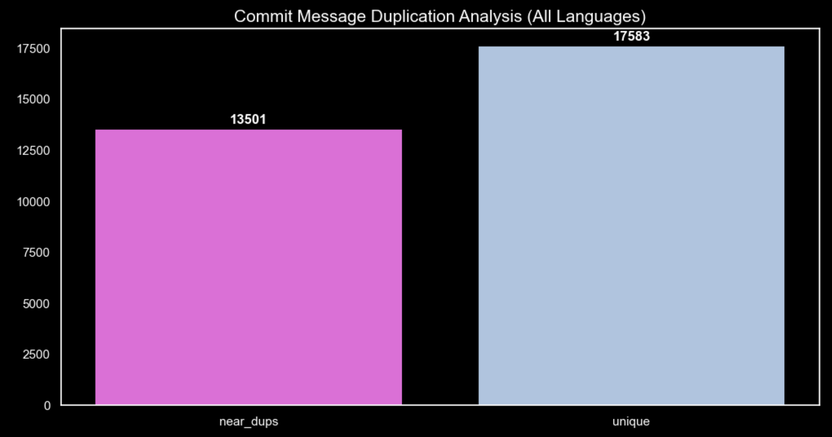
* This section as been sampled down due to time and resource constraints

THE UNIVERSITY OF CHICAGO
Vincent Caldwell December 2nd, 2023

Text Similarity Analysis* [Continued]

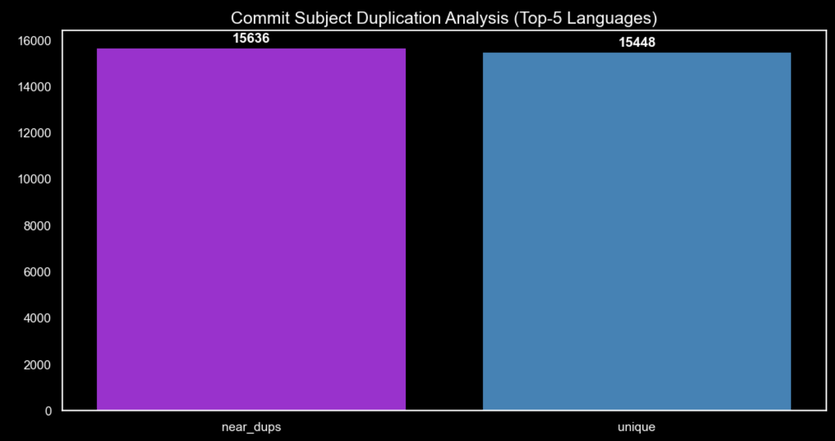
“Subject” and “Message” Duplication Across All Programming Languages

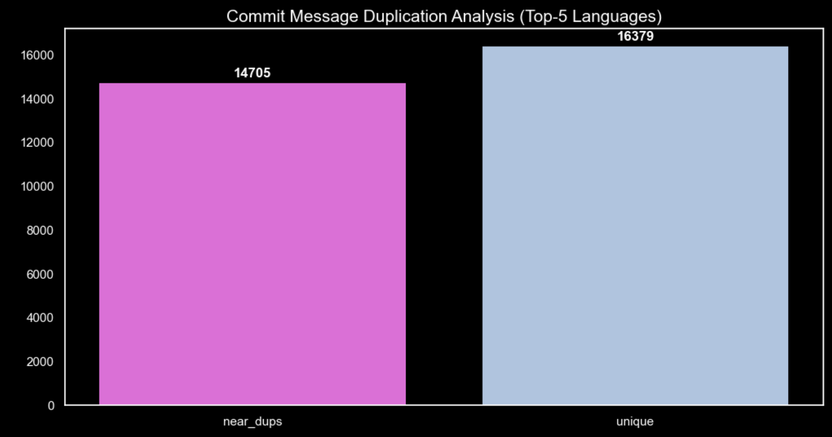




Conclusion: Upon employing a Jaccard distance threshold of 0.3, our analysis reveals a consistent ratio of duplications in both the "subject" and "message" fields. This outcome is logically coherent, given the absence of null values in the "subject," "message," or "language_type" attributes. Additionally, the integration of the programming language table with the commit table exhibited a similar absence of null values following the join operation.

“Subject” and “Message” Duplication Across Top-5 Languages[1]





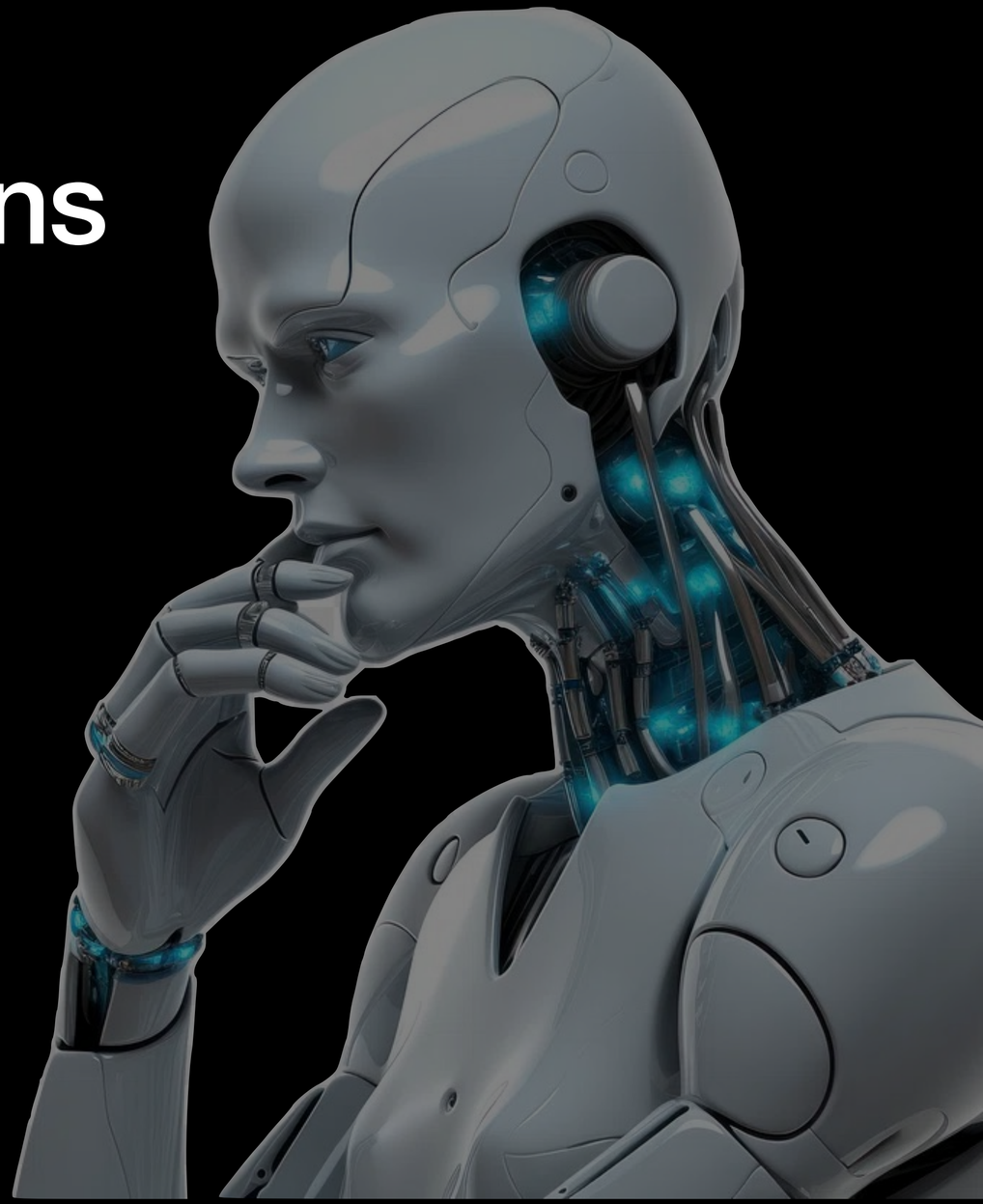
Conclusion: The higher number of duplicate subjects (15,636) compared to duplicate messages (14,705) may indicate that developers are using template-like or standardized commit subjects. This could be a common practice to maintain consistency and provide a brief summary of the commit's purpose.

It's worth noting that the effectiveness of LSH and Jaccard similarity depends on the characteristics of the data and chosen parameters.

* This section as been sampled down due to time and resource constraints

[1]Top-5 Languages are defined as Shell, Python, C, C++, and Perl

Conclusions and Actionable Recommendations



Can these AI solutions provide meaningful improvement in developers' productivity and eventually replace human software engineers and data scientists?

AI solutions and data-driven approaches enhance developer productivity through informed decision-making and optimizing workflows. While AI can streamline certain aspects, the creativity, critical thinking, and problem-solving skills of human software engineers and data scientists remain irreplaceable for nuanced tasks and innovative solution development. Our recommendations are:

Actionable Recommendations:

- Prioritize genuine, technology-driven repository growth, avoiding artificial inflation methods.
- Encourage language diversity and open-source collaboration.
- Keep abreast of language and licensing trends to maintain competitive and modern development practices.
- Continue leveraging AI and machine learning models like LDA for insights while embracing the unique expertise human talent offers.
- Enhance developer toolkits with AI solutions for repetitive and data-intensive tasks, allowing humans to focus on strategic and complex problem-solving.

THE UNIVERSITY OF CHICAGO
Vincent Caldwell December 2nd, 2023

End of Presentation.

Big Data Platforms

Vincent Caldwell December 2nd, 2023