DS Project Document

Team VII

Name	Section	Bench No.
Essam Wisam	2	2
Mohamed Saad	2	15
Mohamed Salama	2	15
Ahmed Waleed	1	13

Supervised by

Dr. Dina ElReedy

Eng. Mohamed Abdullah

May 2023

Table of Contents

Introduction..... 3

Introduction

Problem Definition

In this project we aim to settle the years-old argument regarding the best programming language and the distinctions among different programming languages which is important given the myriad of choices available and the necessity of making a choice in software firms. We consider factors such as success, community engagement, activity and endangerment by studying the stars, pull requests, commit activity and archivals of the corresponding repositories for programming languages.

In particular, we consider the following set of questions:

Descriptive:

1. What is the distribution of the average number of stars over different programming languages? What is the distribution of the average number of stars over time for the top 7 languages in stars?

2. What is the distribution of the most common primary language over time? And what is the distribution of the underlying margin compared to the next most frequent programming language? # Essam Wisam

3. What is the fraction of repositories without a license? What fraction of those with licenses also have a code of conduct?

4. What are the top three sets of programming languages used together over time? What are some strong association rules that can be drawn from these sets?

5. What is the average size of repositories over time for the top 3 programming languages in terms of pull requests?

Exploratory:

6. What is the correlation between the number of watchers and number of pull requests for each programming language?

7. What databases tend to occur more often with different backend frameworks? What front-end frameworks tend to occur most often with the three most frequent backend frameworks?

8. Is there any association between the number of main branch commits, stars and pull requests, and the primary programming language used in a project?

Essam Wisam

9. What licenses are associated with what primary programming languages and project sizes?

Inferential:

10. Does the popularity of Javascript generalize to prove that dynamically typed languages are more popular than statically typed ones?

11. If <X> is the language with the most archived repositories that don't allow forking for the period before 2009 and after 2015, then does this generalize to the whole period (2009 to 2023) regardless whether forking is allowed or not?

Predictive:

12. What is the expected number of pull requests over all Python repositories for the year 2023?

13. What programming language is expected to have the most repos archived in 2023?

Essam Wisam

Folder Structure

```
.
├── DataFiles
│   ├── dataset.csv
│   ├── test.csv
│   ├── train-val.csv
│   ├── train.csv
│   └── val.csv
├── DataPreparation
│   ├── DataPreperation.ipynb
│   ├── Preprocess.py
│   └── Visualize.py
├── Questions
│   ├── D - Language Commonality
│   │   ├── LanguageCommonality.ipynb
│   │   └── Logic.py
│   ├── D - Language Success
│   │   ├── LanguageSuccess.ipynb
│   │   └── Logic..py
│   ├── D - License Prevalence
│   │   ├── LicensePrevalence.ipynb
│   │   └── Logic.py
│   ├── D - Size & Contribution Effect
│   │   ├── SizeAndContributionEffect.ipynb
│   │   └── Logic.py
│   ├── E - Arbitrary Language Predictors
│   │   ├── ArbitraryLanguagePredictors.ipynb
│   │   └── Logic.py
│   ├── E - Contributions&Watchers
│   │   ├── Contributions&Watchers.ipynb
│   │   └── Logic.py
│   ├── E - Database & Frameworks Correspondence
│   │   ├── Databases&Frameworks.ipynb
│   │   └── Logic.py
│   ├── E - Language Associations
│   │   ├── Language Associations.ipynb
│   │   └── Logic.py
│   ├── E - Licenses, Language & Size
│   │   ├── Licenses, Language & Size.ipynb
│   │   └── Logic.py
│   ├── I - Generalizing Archival Trends
│   │   ├── GeneralizingArchivalTrends.ipynb
│   │   └── funs.py
│   ├── I - Generalizing Dynamic Typed Trends
│   │   ├── GeneralizingDynamicTypedTrends..ipynb
│   │   └── Logic.py
│   ├── P - Expected Language Archivals
│   │   ├── ExpectedLanguageArchivals.ipynb
│   │   └── Logic.py
│   └── P - Expected Python Contributions
│       ├── Expected Python Contributions.ipynb
│       └── Logic.py
├── Reports & Dashboard
└── utils.py
```

Data Preparation

Data preparation involves reading the data and putting in a suitable form. Our data preparation module was used for all questions and supported the following:

- Reading specific splits of the data
- Reading specific columns of the data (by name or type)
- Breaking down composite columns
- Deleting useless columns
- Handling missing values by…. # Saad
- Handling outliers by multiple imputation
- Extracting time features

Generic Data Exploration

Dataset Basics

Feature	stars	forks	watchers	isArchived	diskUsageKb	pullRequests
Type	Numerical	Numerical	Numerical	Binary	Numerical	Numerical
Uniques	2033	1156	479	2	39921	1452
Missing	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Outliers	9.45%	7.32%	6.19%	4.97%	13.19%	11.45%

Feature	createdAt	defaultBranch CommitCount	license	code OfConduct	languages Used	languagesSizes	isLicense
Type	Date	Numerical	Categorical	Categorical	Composite	Composite	Binary
Uniques	209558	4415	44	8	37808	164696	2
Missing	0.0%	0.0%	47.0%	96.0%	0.0%	0.0%	0.0%
Outliers	0.0%	8.77%	0%	0%	0%	0%	0%

Descriptive Statistics

	stars	forks	watchers	isArchived	diskUsageKb	pullRequests	defaultBranch CommitCount
count	210001	210001	210001	210001	210001	210001	210001.
mean	76.61	20.54	7.09	0.05	24673.71	24.45	645.70
std	1021.24	206.20	35.77	0.22	269947.32	314.63	17676.89
min	3.00	0.00	0.00	0.00	0.00	0.00	-1.00
25%	7.00	1.00	2.00	0.00	96.00	0.00	9.00
50%	12.00	4.00	3.00	0.00	518.00	1.00	27.00
75%	30.00	11.00	6.00	0.00	4715.00	6.00	89.00
max	264811.00	27028.00	5923.00	1.00	41796529.00	81258.00	1154219.00

Insights

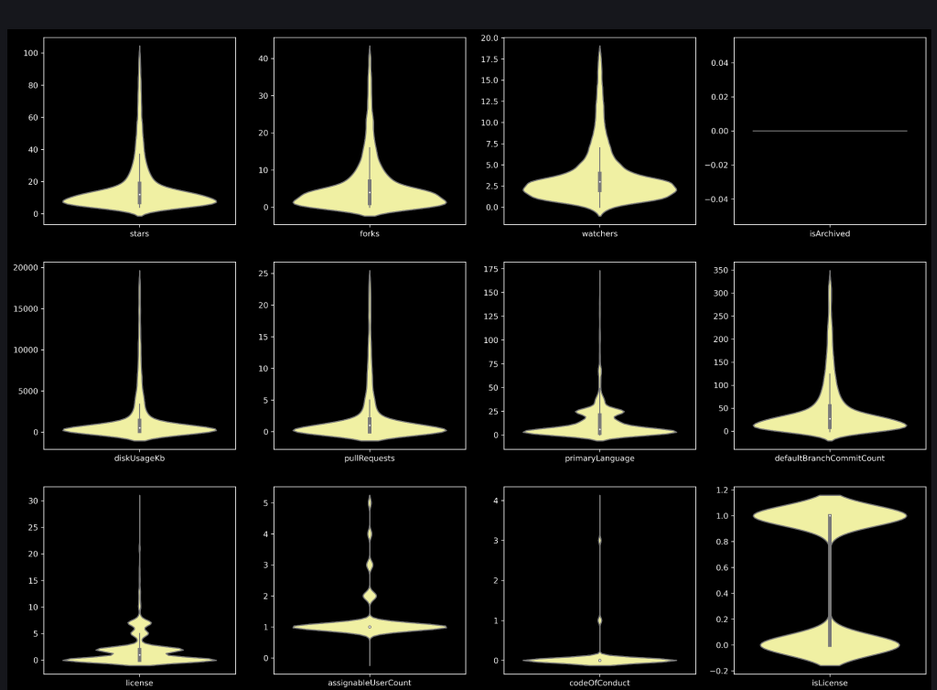
✦ Most max values are far beyond the mean and median which signals outliers (removed after this)

✦ 50% of repos have 12 stars or less, 4 forks or less, 3 watchers or less, 1 pull request or less, 27 commits or less

✦ Compared to other features, stars vary more widely from repo to repo, followed by pull requests and forks

✦ The maximum stars figure is realistic and forks can be realistic given that clones trigger them

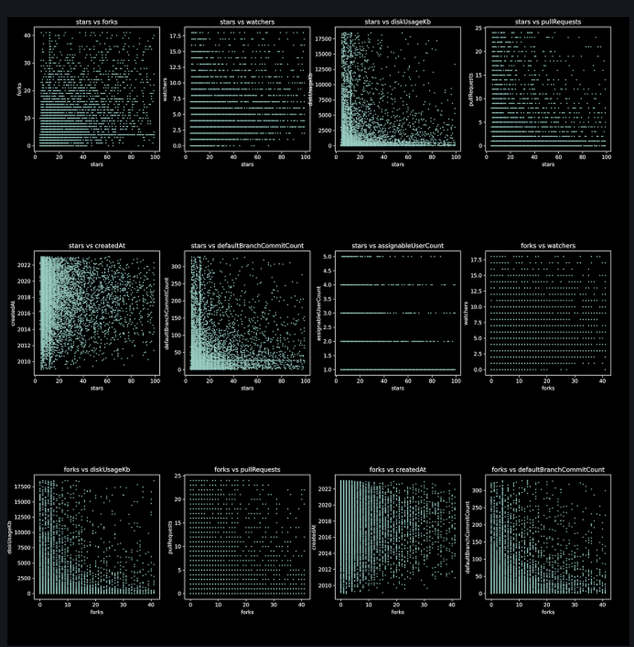
Feature Distributions

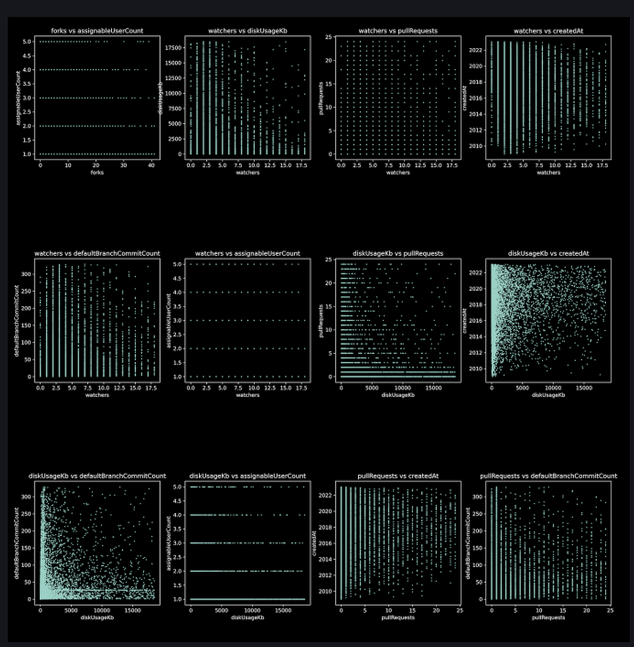


Insights

- ✦ The distributions seem to all be left-skewed; closely forming a chi-square distribution
- ✦ The distributions for license and programming language are multimodal but still skewed towards specific categories
- ✦ The random sample for 10K rows used for the plots may have not included any archived repos which indicates its scarcity

Pairwise Numerical Patterns

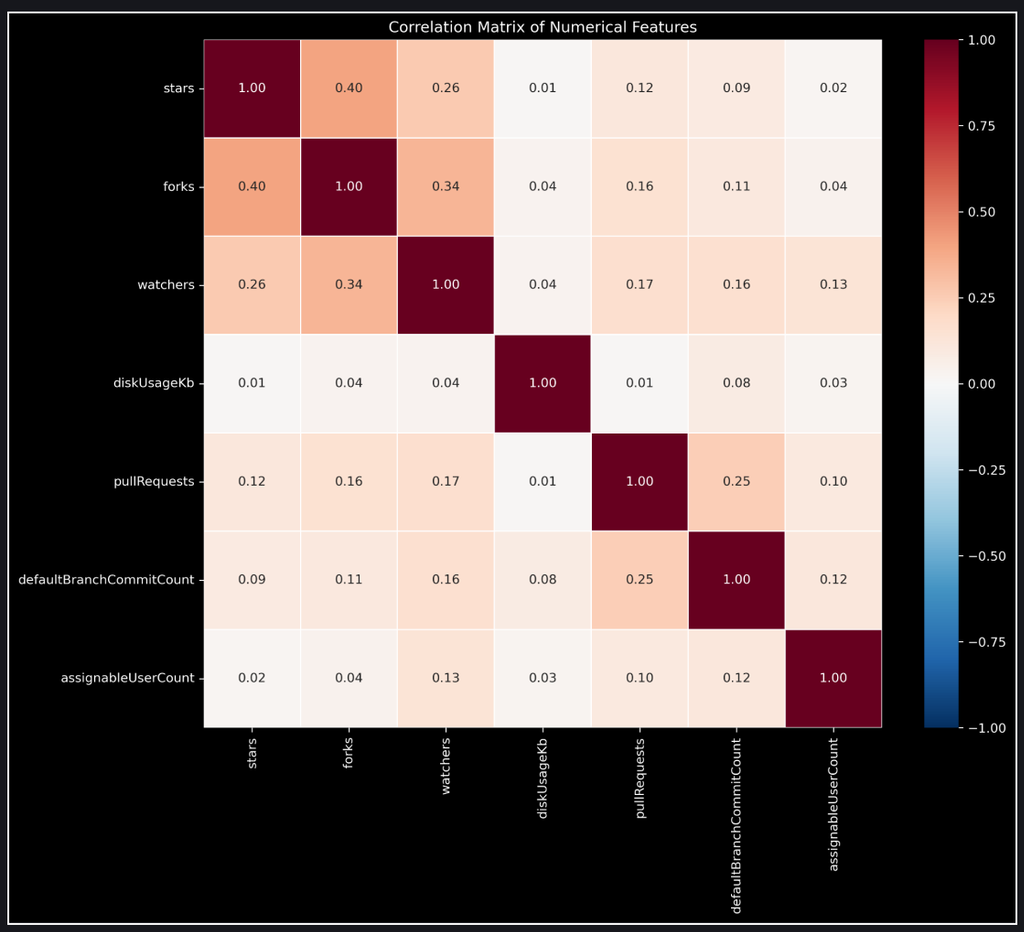




Insights

- ✦ Most relationships seem complex (e.g., not linear)
- ✦ Stars/DiskUsage and Stars/Commits seem to be anti-correlated. Likewise for Forking/Commits
- ✦ Newer repos seem to have more variance over most features

Studying Numerical Correlations



Insights

- ✦ Mostly no correlation between the variables
- ✦ Relevant positive correlation between stars and forks/watchers and forks
- ✦ Slight negative correlation between watchers and createdAt

Questions

Language Commonality

Stating Questions

- ✦ What is the distribution of the most common primary language over time?
- ✦ What is the distribution of the underlying margin compared to the next most frequent programming language?

Set Expectations	Collect Data	Match Expectations & Data
The question is interesting and answerable.	It is interesting because the developers need to know what language to learn. It is answerable because the data is available and we can use it with some statistical methods to answer the question.	Data and expectations match.

Exploratory Data Analytics

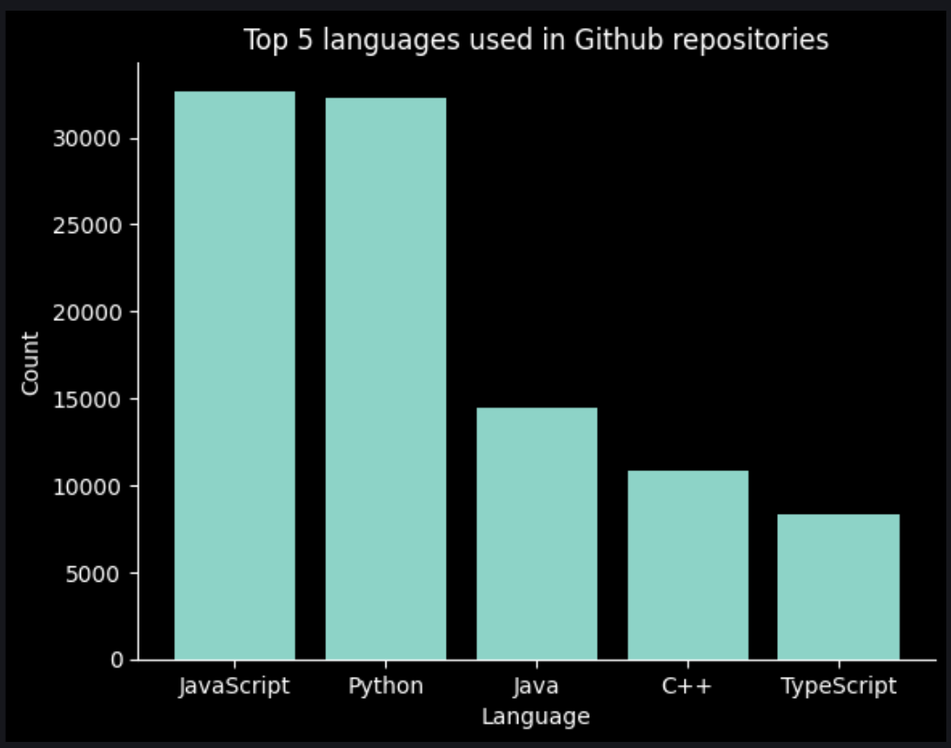
Let's explore the available primary programming languages in the projects present in the dataset

```
Number of available languages:  377

Sample languages:
C          8017
HTML       6732
PHP        8246
Python    32295
C++       10835
```

Set Expectations	Collect Data	Match Expectations & Data
There is many used programming languages in github.	already found 377 unique languages in the dataset.	expectations and data match.

Model Building



The graph shows the primary programming languages that have been most commonly used from 2009 to 2022. Additionally, similar graphs have been created for each individual year. Then we can determine the most prevalent programming language for each year and calculate the margin between it and the second most commonly used language.

Set Expectations	Collect Data	Match Expectations & Data
The model we be able to get most used language and the margin between it and the next most used language.	Model got most used language and the margin between it and the next most used language.	expectations and data match.

Result Interpretation

Most common primary languages used in Github repositories and the margins between them

- Earlier in 2009 and 2010 the most common primary language used in Github repositories was Ruby
- From 2011 to 2017 javascript was the most common primary language used in Github repositories
- From 2018 to 2022 python is the most common primary language used in Github repositories
- The most common primary language used in Github repositories over all years is JavaScript with 32684 repositories using it.
- The Second most common primary language used in Github repositories over all years is Python with 32295 repositories using it.
- As seen they are almostly equally used with a margin of 389 repositories.
- The Third most common primary language used in Github repositories over all years is Java with 14489 repositories using it.
- As seen it is used about half as much as the second most used language with a margin of 17806 repositories.

Set Expectations	Collect Data	Match Expectations & Data
Python and javascript are the most used languages in github over all years, with large margin between the next language after them, Currently python is the most used language in github, Javascript is also used a lot in github.	results as shown (already found that python and javascript are the most used languages in github , with large margin between the next language after them and python is the most used right now)	expectations and data match.

Communicating Results

Insights about the most common primary languages used in Github repositories

- As Javascript is the most important language for web development, it is the most used language in Github repositories over all years.
- As python is used for many purposes such data science, machine learning and more, it is the second most used language in Github repositories.
- As AI is currently the most important field in computer science and the main language used in AI is python, it is the most used language right now.
- The margin between python and javascript over all years is very small it is only 389 repositories
- But the margin to the third most used language is very large, it is 17806 repositories
- Earlier in 2009 and 2010 the most common primary language was Ruby as it was the most important language for web development at that time
- From 2011 to 2017 javascript was the most common primary language used in Github repositories as web development has become more important
- From 2018 to 2022 python is the most common primary language used in Github repositories as AI has become more important

Set Expectations	Collect Data	Match Expectations & Data
The professor will be interested in knowing these results.	As written above.	Will be known after the presentation.

Language Success

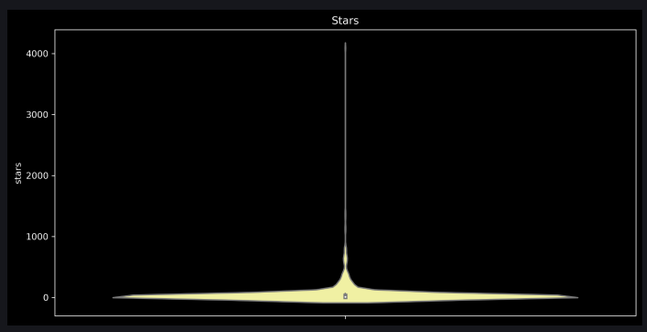
Stating Questions

- What is the distribution of the total number of stars over different programming languages?
- What is the distribution of the average numbe of stars over time for the top 7 languages?
- What is the distribution of the number of the normalized stars overtime for the top 7 languages?

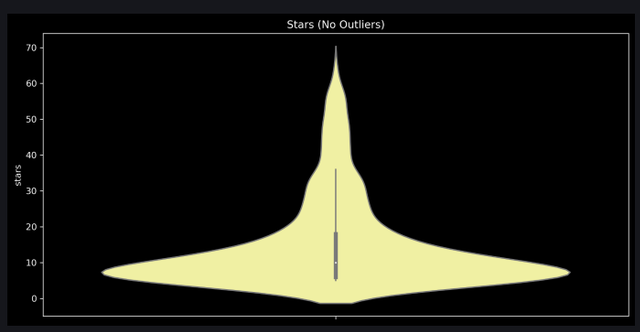
Set Expectations	Collect Data	Match Expectations & Data
Question is specific, interesting and novel.	Clearly the question is specific as it is all about a simple data quries. The question is interesting as it is directly related to the popularity of programming languages. The question is already answered in the line chart form in this site A SMALL PLACE TO DISCOVER LANGUAGES IN GITHUB	Change it to a novel one that is not giving a number of star but the number of stars normalized by the number of repositories.

Exploratory Data Analytics

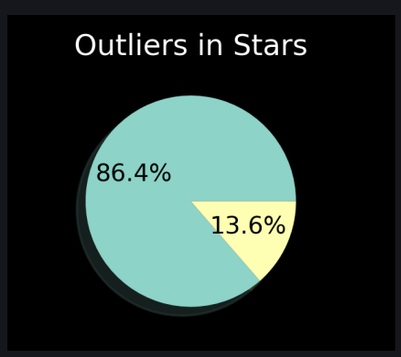
The distribution of stars without outliers removal



After removing the outliers



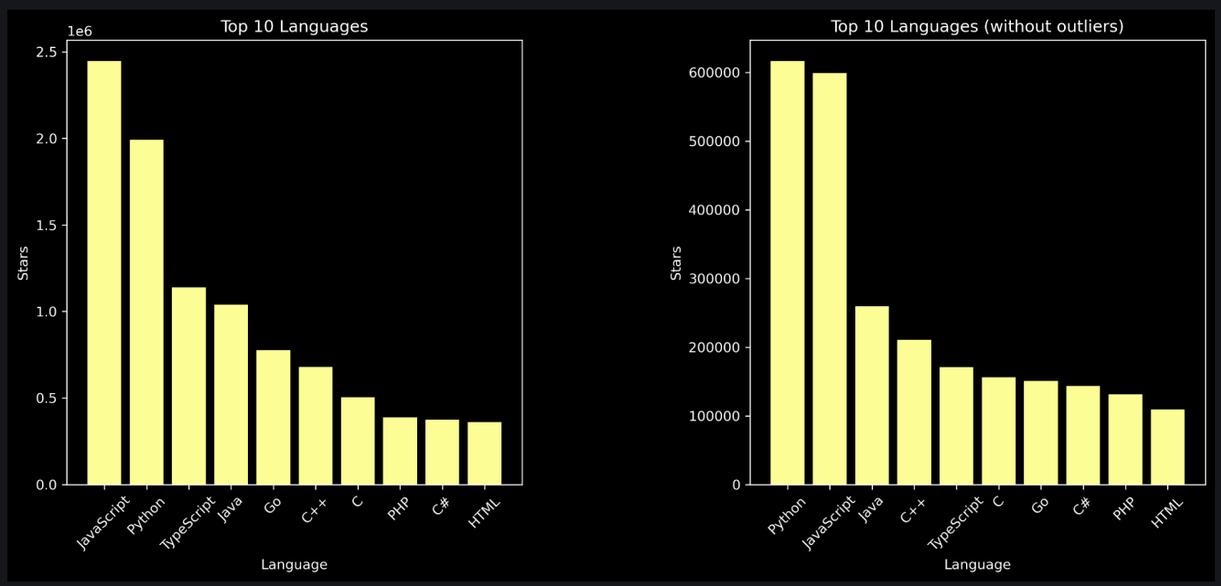
Outliers ratio in stars



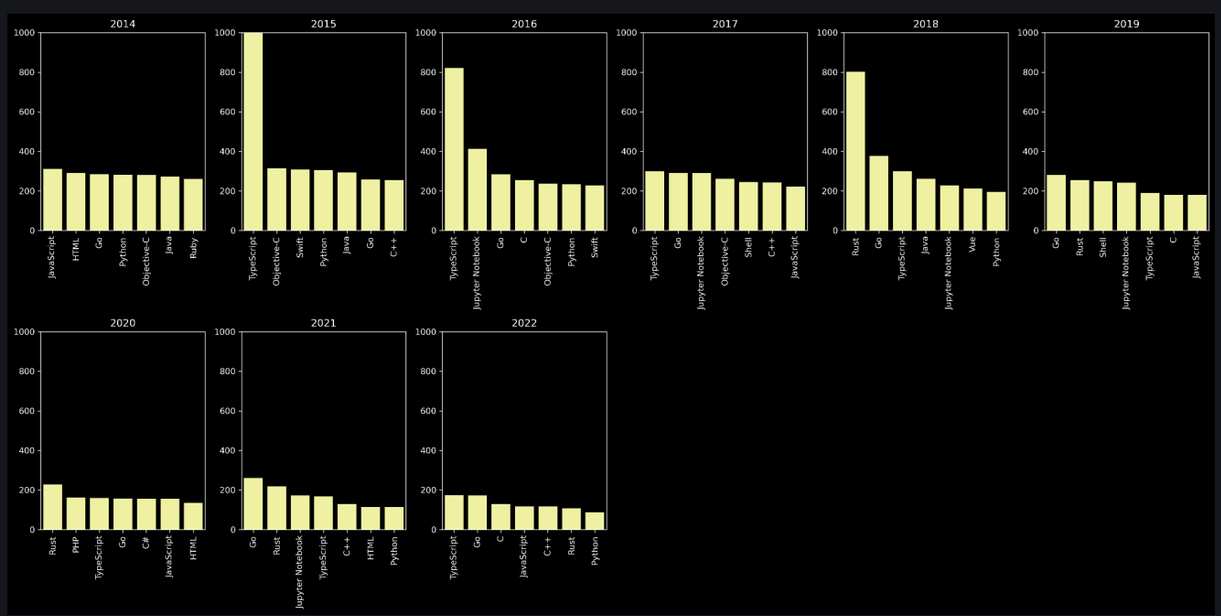
Set Expectations	Collect Data	Match Expectations & Data
There are extreme outliers in stars columns and most of the values are close to zero as github is filled with small projects from beginners.	From the distribution of the data shown above (as viloin plots), we can see that distribution of the data is centered close zero and there are extreme outliers in stars column.	The data matches the expectations.

Model Building

Top 10 languages regarding the total number of stars [with and without outliers]



The Top 7 languages for each year regarding the normalized star value (stars / number of repositories)



Set Expectations	Collect Data	Match Expectations & Data
The plot of total stars per language and the distribution of the stars over time will help to understand the relation between the number of stars and the programming language.	The plot of total stars per language and the distribution of the normalized stars over time are shown above.	The current model is good enough to explain the data.

Result Interpretation

Insights

- ✦ The most starred language is JavaScript followed by Python but that doesn't take into account the number of repositories per language.
- ✦ TypeScript is the most appeared language in the 1st place regarding the normalized stars which reflects the success of the language.

Set Expectations	Collect Data	Match Expectations & Data
The total number of stars per language and the distribution of the stars over time will help to understand the relation between the number of stars and the programming language is going to answer the question.	Shown in the insights section.	The question is answered.

Communicating Results

Insights

- ✦ Typescript had and still a huge love from the developers
- ✦ The trending language is continuously changing

Set Expectations	Collect Data	Match Expectations & Data
Professor & TA will appreciate and be interested in the communicated information.	As above.	Will be known after the presentation day.

License Prevalence

Stating Questions

- ✦ What is the fraction of repositories without a license ?
- ✦ What fraction of those with licenses also have a code of conduct ?

Set Expectations	Collect Data	Match Expectations & Data
Question is interesting, specific, answerable and novel.	Indeed, its answerable just by using simple statistics and its interesting because it conveys the degree to which developers have to be made more aware of the importance of licensing and code of conduct. It's clearly specific. It's novel because the most recent publicly known statisitc regarding this comes from 2018	Date and exepctations match to a decent extent.

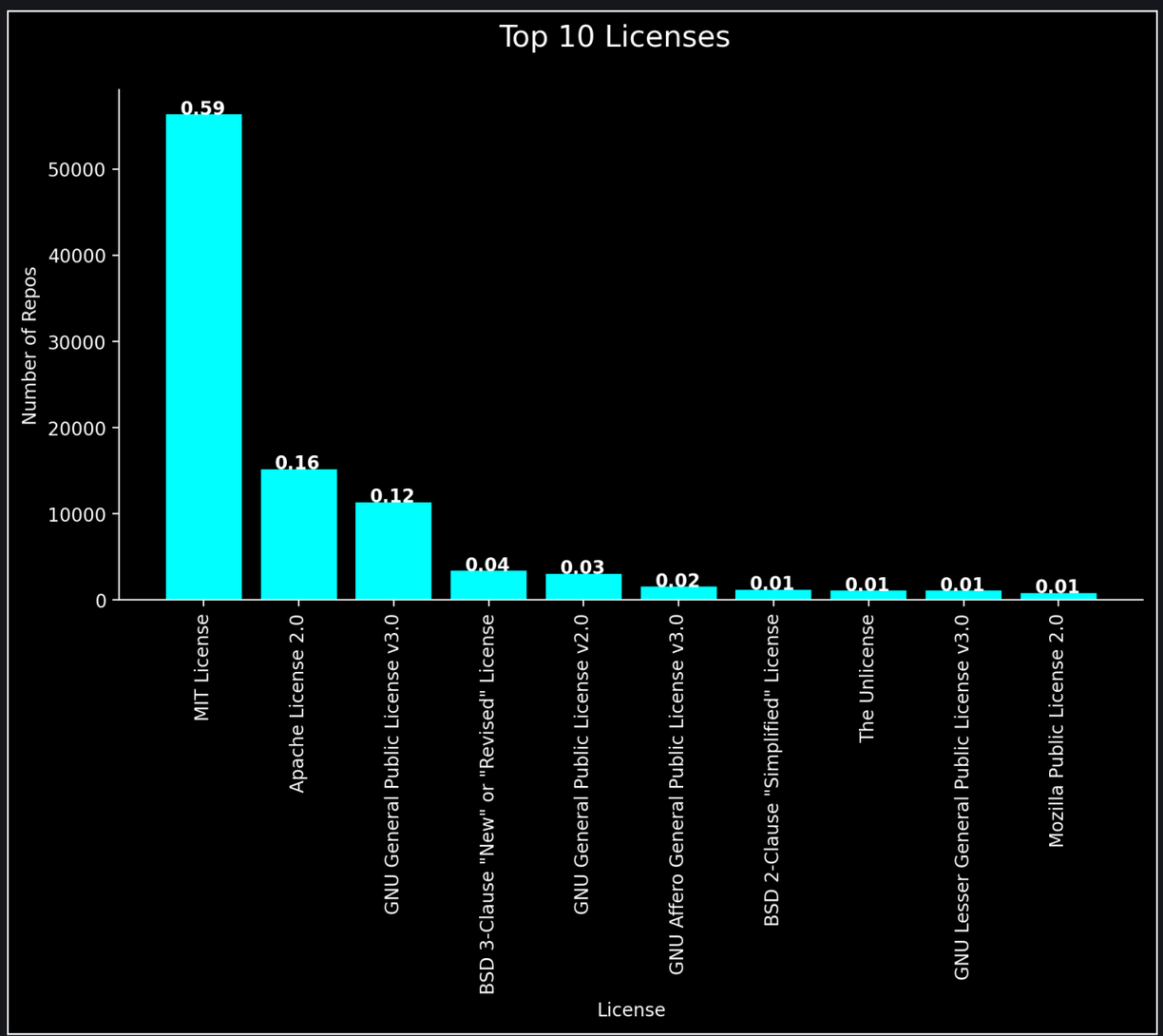
Exploratory Data Analytics

Let’s explore the available licenses in the projects present in the dataset

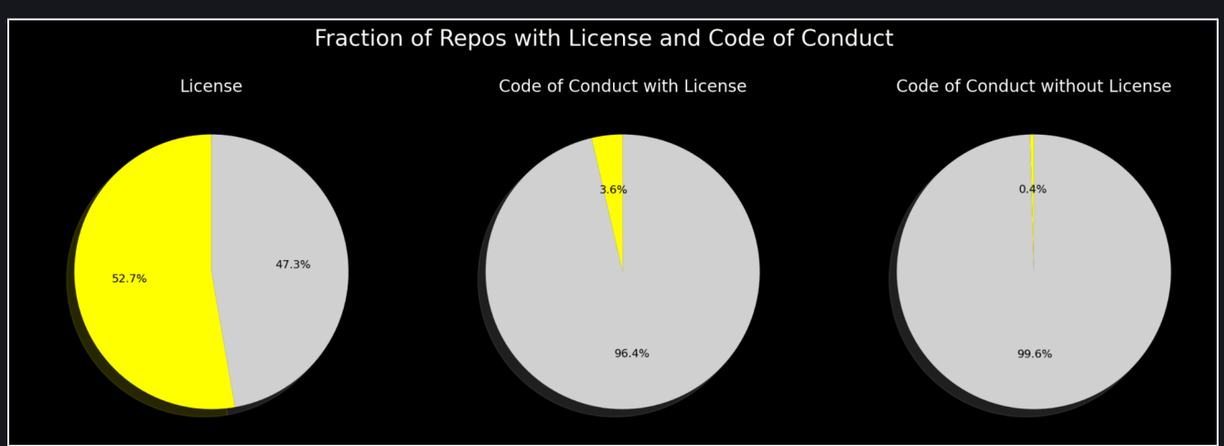
Set Expectations	Collect Data	Match Expectations & Data
There are over 80 open-source licenses and the dataset is expected to cover a significant variety of them.	Indeed, as per the exploratory analysis, it has over 40 licenses.	Expectations and data match.

Let's see what are the most common licenses.

Set Expectations	Collect Data	Match Expectations & Data
As background knowledge, we know that MIT Licesnce is the most popular.	The output from the model is as shown.	Indeed, they match.



Model Building



Set Expectations	Collect Data	Match Expectations & Data
The model will be able to use basic statistics to provide a valid answer to the question.	The output from the model is as shown.	Upon examining the model output, high-level expectations match the data. There are many repos with no license and even far less with a code of conduct.

Result Interpretation

Set Expectations	Collect Data	Match Expectations & Data
Many repos with no license as by default Github does not add a license to a repo and most developers are not entirely aware of the differences between licenses and their importances.	Results are as shown above.	They match as per the insights.

License Prevalence Results

✦ The majority of repos do have a license at 52.7%

✦ But there is a significant amount of repos that do not have a license at 47.3%

✦ The most common license is the most permissive one which is good news for FOSS development.

✦ The second most common, is also a permissive one but only requires modifications to be stated and is more legally suitable.

Set Expectations	Collect Data	Match Expectations & Data
Code of conduct is expected to be rare as we as developers do not remember seeing one ever. It's natural to expect that if a developer is not aware of the importance of a license, they would not be aware of the importance of a code of conduct.	Results are as shown above.	They match as per the insights.

Code of Conduct Results

✦ The absolute majority of repos do not a code of conduct at 1 - 0.036*0.52+0.004*0.473 = 0.98

✦ Still, knowing that a repo has a license rather than not multiplies the chance of having CoC by 3.6/0.4 = 9 times

Communicating Results

License Prevalence

✦ Although repos don't require a license by default, about one in every two repos has a license.

✦ It means that about half of the repos are protected by the default strict copyright law. Without explicit permission, only viewing the code is legal.

✦ This does not encourage FOSS development, one of the main goals of GitHub.

Code of Conduct

✦ The absolute majority of repos do not have a code of conduct; there is no option to add it at the time of repo creation.

✦ Although, to organize collaboration and interaction in a community, a code of conduct is

important; it's only used for a tiny fraction of repos.

✦ Github may encourage using it by adding it as an option at the time of repo creation.

Set Expectations	Collect Data	Match Expectations & Data
Professor & TA will appreciate and be interested in the communicated information.	As above.	Will be known after the presentation day.

Size & Contribution Effect

Stating Questions

What is the average size of repositories over time for the top 3 programming languages in terms of pull requests?

Set Expectations	Collect Data	Match Expectations & Data
This question is answerable and it's interesting.	There is no existent answer for such question. It reflects the relationship between contriubtions and size of the repository.	Both match!

Exploratory Data Analytics

Number of primary programming languages

✦ there are 401 primaryLanguages

+ Code + Markdown

Set Expectations	Collect Data	Match Expectations & Data
I expect that there are more than 250 programming languages in the dataset.	Indeed, there are 401 different primary languages in the dataset.	Expectation agrees with data.

Model Building

	language	total pullRequests	average diskUsageKb
0	JavaScript	1149807	836.031047
1	TypeScript	985966	356.830300
2	Python	866438	404.479674

Set Expectations	Collect Data	Match Expectations & Data
It's expected that the model will return the top 3 programming languages with their corresponding repositries' avg. sizes.	The top 3 programming languages with their corresponding repositries' avg. sizes are shown in the table above.	Results seems resonable.

Result Interpretation

Results

✦ JavaScript, TypeScript, and Python are the top 3 programming languages used in terms of pull requests

✦ Sizes are correlated with pull requests to some extent

Set Expectations	Collect Data	Match Expectations & Data
I expect JavaScript to be at least in the top 3 programming languages. As it's has a lot of usage in web development. Also I expect to see a correlation between the number of pull requests and the size of the repository.	Data is shown above	Expectation agrees with data.

Communicating Results

Results

✦ The open source community is dominated by JavaScript, TypeScript, and Python

✦ TypeScript actually prevalence rate is promising as it is a relatively new language but it is already the second most used language in the open source community

✦ Sizes are correlated with pull requests to some extent

Set Expectations	Collect Data	Match Expectations & Data
Professor & TA will appreciate and be interested in the communicated information.	As above.	Will be known after the presentation day.

Arbitrary Language Predictors

Stating Questions

✦ Is there any association between the number of branch commits, number of stars, number of pull requests and the primary programming language used in the project?

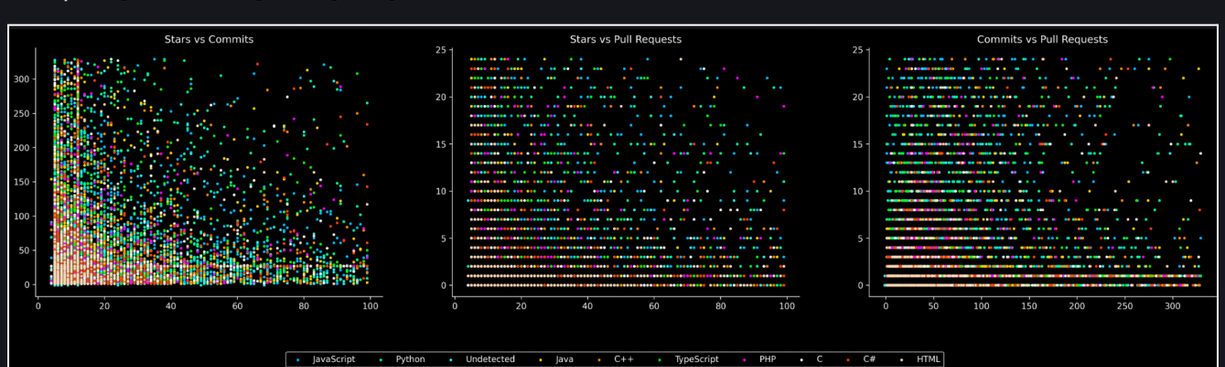
Set Expectations	Collect Data	Match Expectations & Data
Question is interesting, specific, answerable and novel.	Question doesn't clearly define whether the association is to be checked among the variables or between each variable and the language.	Rephrase the question to make it more specific. In this, it will check the association between each variable and the language.

✦ Is there any association between any of the the number of branch commits, number of stars, number of pull requests and the primary programming language used in the project?

Set Expectations	Collect Data	Match Expectations & Data
Question is interesting, specific, answerable and novel.	Interesting because it sheds light on how languages are different in terms of community engagement and successful projects. Answerable using Pearson's correlation, correlation ratio and hypothesis tests. Novel because its not a common fact that these factors are really different across languages.	Date and exepctations match to a decent extent.

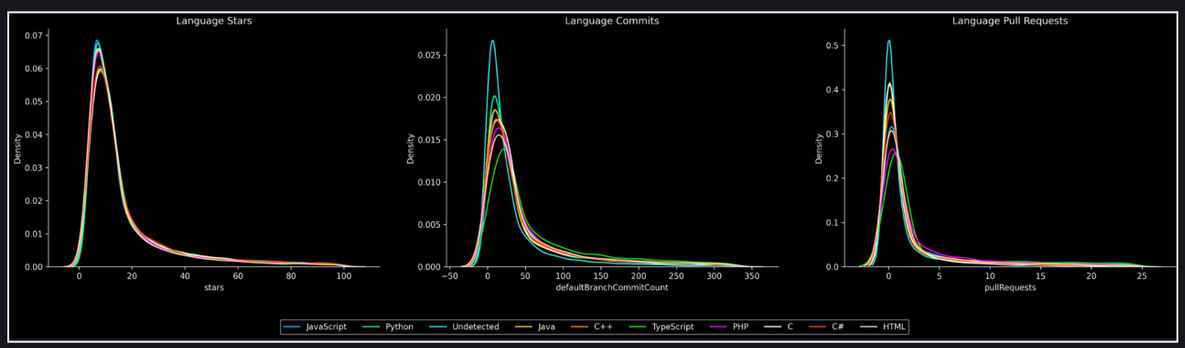
Exploratory Data Analytics

Let's see if there a precise distinctive association between the three factors and the programming language



Set Expectations	Collect Data	Match Expectations & Data
It would be very odd to find that the specific number of commits, stars and pull requests are strongly predictive of the language used in the project.	Data is as shown	Indeed, it's too hard to find any clue of separation among languages using these variables.

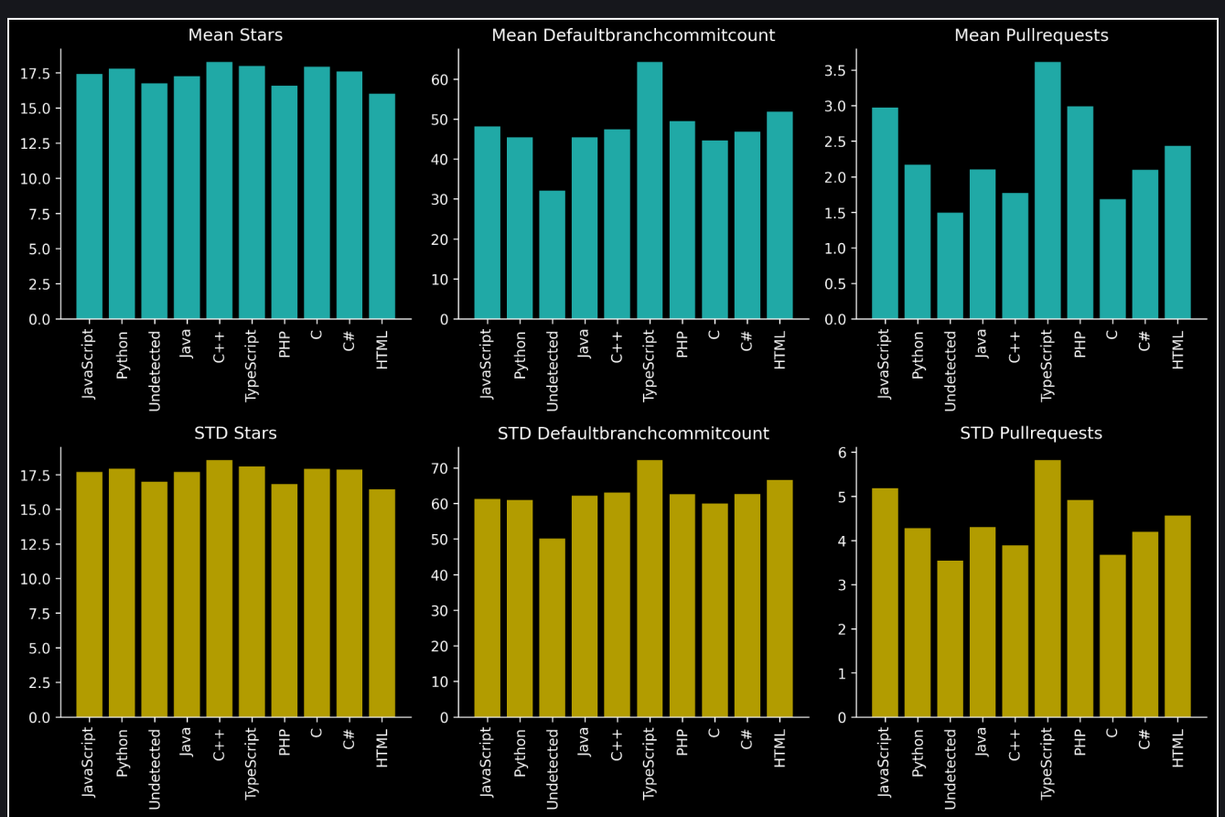
Let’s confirm this using densities



Set Expectations	Collect Data	Match Expectations & Data
No anomalies regarding the distribution of stars, commits and pull requests.	Data is as shown	Indeed, they are very close to each other.

Set Expectations	Collect Data	Match Expectations & Data
The document (undetected) category should not involve engagement, activity or success more than other categories.	Data is as shown	Indeed, it is the most left-skewed in all three aspects (pull requests, commits and stars).

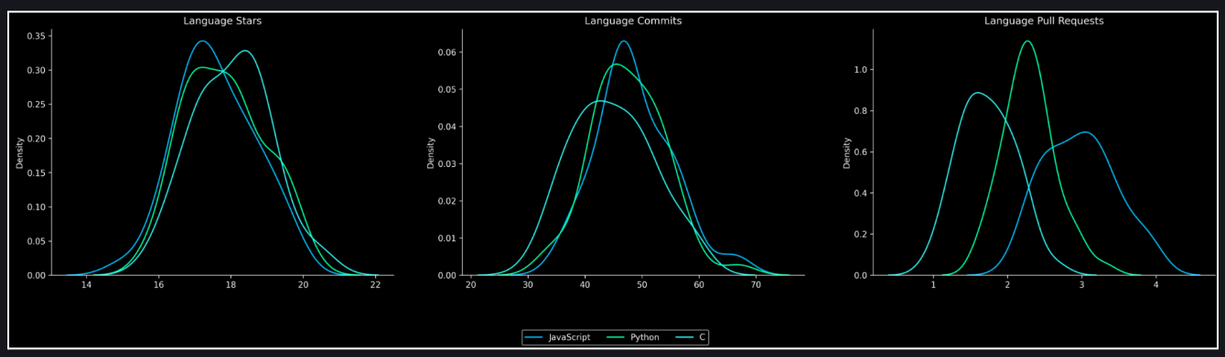
Let’s see if there at least a high-level distinctive association



Set Expectations	Collect Data	Match Expectations & Data
There should at least be a high-level association among languages and the number of commits, stars and pull requests.	Data is as shown	For stars, there is a low association however, for activity and engagement, different languages tend to be different.

Model Building

Let’s confirm normality as needed by the t-statistic



Hypothesis test regarding the low association between languages and stars

Lets test the following Hypothesis:

H_0: It doesn't hold that $16 < \mu^l_{stars} < 19$ for one or more languages l.

H_1: It holds that $16 < \mu^l_{stars} < 19$ for each language l.

We will set the significance level $\alpha = 0.05$ and perform two one-sided tests for each language.

Results

JavaScript	Python	Undetected	Java	C++	TypeScript	PHP	C	C#	HTML
Accept: μ - 15.5 > 0	Accept: μ - 15.5 > 0	Accept: μ - 15.5 > 0	Accept: μ - 15.5 > 0	Accept: μ - 15.5 > 0	Accept: μ - 15.5 > 0	Accept: μ - 15.5 > 0	Accept: μ - 15.5 > 0	Accept: μ - 15.5 > 0	Accept: μ - 15.5 > 0
Accept: μ - 19.5 < 0	Accept: μ - 19.5 < 0	Accept: μ - 19.5 < 0	Accept: μ - 19.5 < 0	Accept: μ - 19.5 < 0	Accept: μ - 19.5 < 0	Accept: μ - 19.5 < 0	Accept: μ - 19.5 < 0	Accept: μ - 19.5 < 0	Accept: μ - 19.5 < 0

Set Expectations	Collect Data	Match Expectations & Data
Due to the sufficiently large sample size, and the results gathered from EDA we expect to reject the null hypothesis	Data is as shown	Indeed, we reject the null hypothesis for all languages.

Hypothesis test regarding the High Commit Activity in TypeScript compared to Other Languages

Lets test the following Hypothesis where TS indicates the TypeScript language:

H_0: It holds that $\mu^{TS}_{Commits} \leq \mu^l_{Commits}$ for one or more language l.

H_1: It holds that $\mu^{TS}_{Commits} > \mu^l_{Commits}$ for each language l.

We will set the significance level $\alpha = 0.05$ and perform a one-sided tests for each language.

Results

JavaScript	Python	Undetected	Java	C++	PHP	C	C#	HTML
Accept: μ(TS) - μ(JavaScript) > 0	Accept: μ(TS) - μ(Python) > 0	Accept: μ(TS) - μ(Undetected) > 0	Accept: μ(TS) - μ(Java) > 0	Accept: μ(TS) - μ(C++) > 0	Accept: μ(TS) - μ(PHP) > 0	Accept: μ(TS) - μ(C) > 0	Accept: μ(TS) - μ(C#) > 0	Accept: μ(TS) - μ(HTML) > 0

Set Expectations	Collect Data	Match Expectations & Data
Due to the sufficiently large sample size, and the results gathered from EDA we expect to reject the null hypothesis	Data is as shown	Indeed, we reject the null hypothesis for all languages.

Hypothesis regarding the Poor Community Engagement in C compared to Other Languages

Lets test the following Hypothesis where C indicates the C language:

H_0: It holds that $\mu^{C}_{PullRequests} \geq \mu^{l}_{PullRequests}$ for one or more language l.

H_1: It holds that $\mu^{C}_{PullRequests} < \mu^{l}_{PullRequests}$ for each language l.

We will set the significance level $\alpha = 0.05$ and perform a one-sided tests for each language.

JavaScript	Python	Undetected	Java	C++	TypeScript	PHP	C#	HTML
Accept: μ(C) - μ(JavaScript) < 0	Accept: μ(C) - μ(Python) < 0	Accept: μ(C) - μ(Undetected) > 0	Accept: μ(C) - μ(Java) < 0	Cannot Reject: μ(C) - μ(C++) > 0	Accept: μ(C) - μ(TypeScript) < 0	Accept: μ(C) - μ(PHP) < 0	Accept: μ(C) - μ(C#) < 0	Accept: μ(C) - μ(HTML) < 0

Set Expectations	Collect Data	Match Expectations & Data
Due to the sufficiently large sample size, and the results gathered from EDA we expect to reject the null hypothesis	Data is as shown	Indeed, we reject the null hypothesis for all languages.

Result Interpretation

Insights

✦ There is no precise association between the number of stars, pull requests, and commits with the language used.

✦ For stars, there almost seems to be no association at all but for the other languages, there is a high-level association.

✦ Over all languages, TypeScript has the most commits per project in the population of projects

✦ Over all languages, C has the lowest pull requests per project in the population of projects

Set Expectations	Collect Data	Match Expectations & Data
There is some association between the number of commits and the number of pull requests and the language because they are used by different communities of developers or purposes	As shown above	Indeed, the association holds but only at a high-level for pull requests and commits besides holding very weekly for stars

Communicating Results

Insights

✦ We cannot predict the language given stars, pull requests, and commits. In other words, no strong or precise association

✦ Languages seem to be equally successful as their stars are not significantly different on average

✦ There is a high-level association for commits; for instance, TypeScript can be regarded as the most active language

✦ There is a high-level association for pull requests; for instance, C (& C++) can be regarded as the least collaborative language

Set Expectations	Collect Data	Match Expectations & Data
Professor & TA will appreciate and be interested in the communicated information.	As above.	Will be known after the presentation day.

Contributions & Watchers

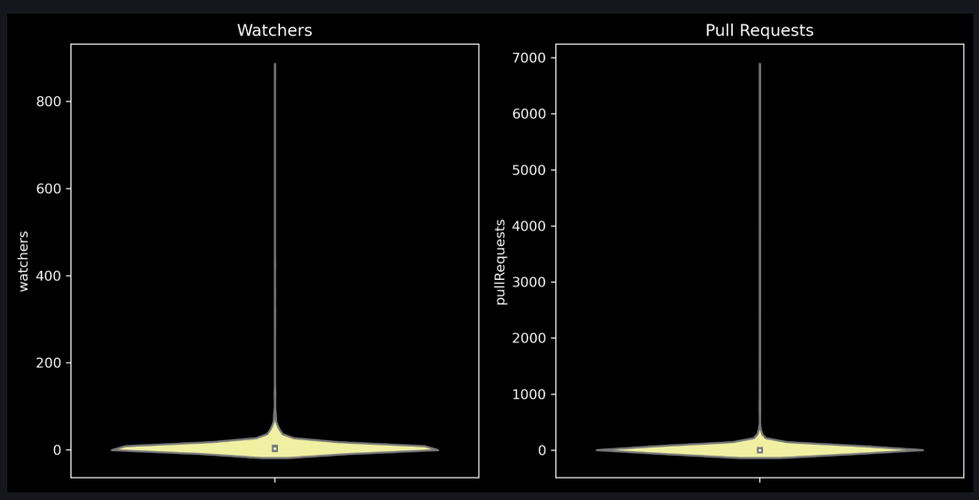
Stating Questions

✦ What is the correlation between the number of watchers and the number of pull requests for each programming language ?

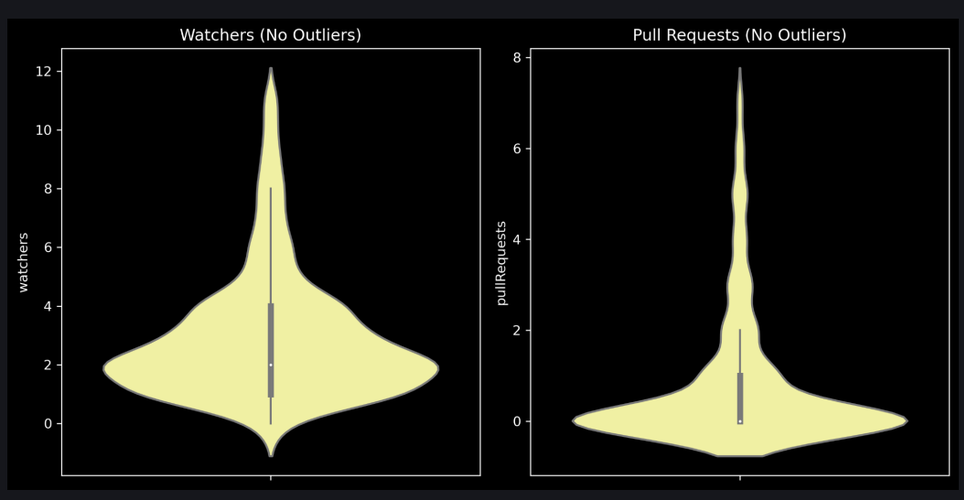
Set Expectations	Collect Data	Match Expectations & Data
The question is specific, interest and novel.	The question is specific using a statistical test. It is interesting as it will help the business in the chioce of the programming language to use in the next project (the question is directly related to the community support). It is novel as it has not been asked before as far as my research goes, and the closest question I found to that is disscussed in this paper Understanding Watchers on GitHub.	The question is indeed specific, interest and novel.

Exploratory Data Analytics

The distribution of watchers and contribution (With outliers)



The distribution of watchers and contribution (Without outliers)



Set Expectations	Collect Data	Match Expectations & Data
There are extreme outliers in watch and contribution columns and most of the values are close to zero as github is filled with small projects from beginners.	From the distribution of the data shown above (as viloin plots), we can see that distribution of the data is centered close zero and there are extreme outliers in watch and contribution columns.	The data matches the expectations.

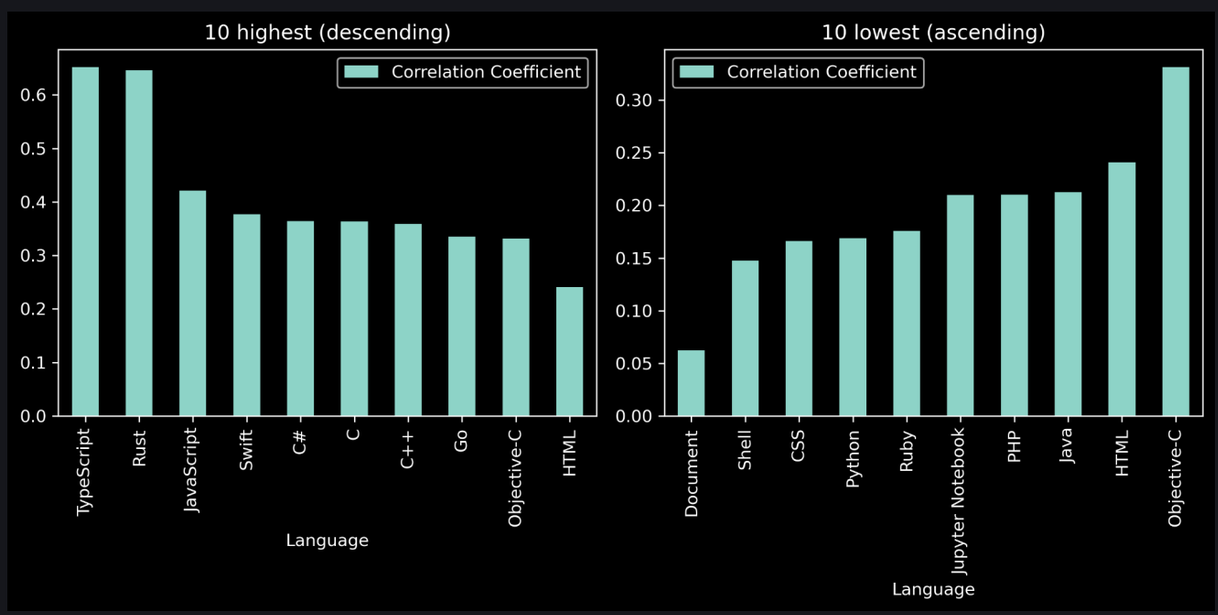
Chi square test to check for the dependance between watchers and contribution

```
dof=693578
probability=0.950, critical=695516.405, stat=17645647.575
Dependent (reject H0)
```

Set Expectations	Collect Data	Match Expectations & Data
With the increase of the number of watchers, the number of pull requests increases.	From the chi-square test, we can see that indeed the number of pull requests increases with the increase of the number of watchers.	The data matches the expectations.

Model Building

The following plot shows pearson correlation between the watchers and the contributors for top programming languages regarding this measure



Set Expectations	Collect Data	Match Expectations & Data
The correlation coefficient will allow us to see the relationship between the number of watchers and the number of pull requests.	From the correlation coefficient, we can see that there are some languages that has a stronger correlation between the number of watchers and the number of pull requests than others (implying that the engagement of the community is higher in those languages).	The model was able to answer the question.

Result Interpretation

Insights

- Typrscript has the highest correlation coefficient between watchers and pull requests
- Document repositories have the lowest correlation coefficient between watchers and pull requests (seems logical)
- The is still a weak correlation between watchers and pull requests for the most famous on github javascript

Set Expectations	Collect Data	Match Expectations & Data
The corelation between the number of watchers and the number of pull requests is understood.	Shown in insights.	The question is answered.

Communicating Results

Insights

- ✦ Typescript seems to has the most engaged community members
- ✦ Surprisingly python came in the lowest 10 list in community engagement
- ✦ Also it is surprising to see C and C++ in the top list

Set Expectations	Collect Data	Match Expectations & Data
Professor & TA will appreciate and be interested in the communicated information.	As above.	Will be known after the presentation day.

Database & Framework Correspondence

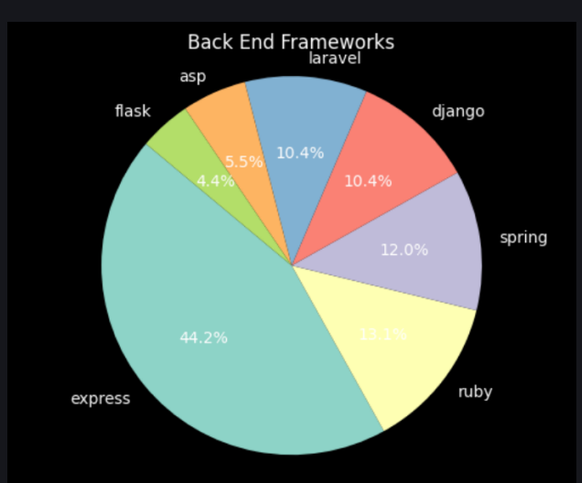
Stating Questions

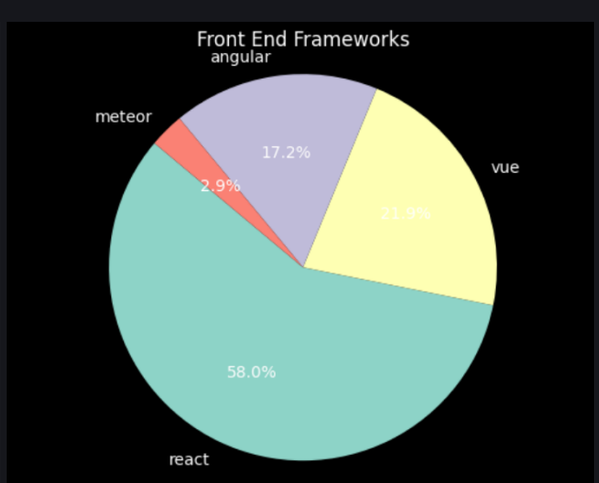
Stating Questions

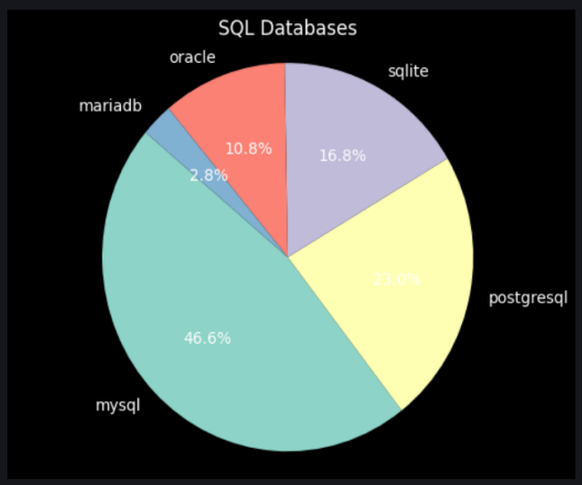
- ✦ What front-end frameworks tend to occur most often with the three most frequent backend frameworks?
- ✦ What databases tend to occur more often with different backend frameworks?

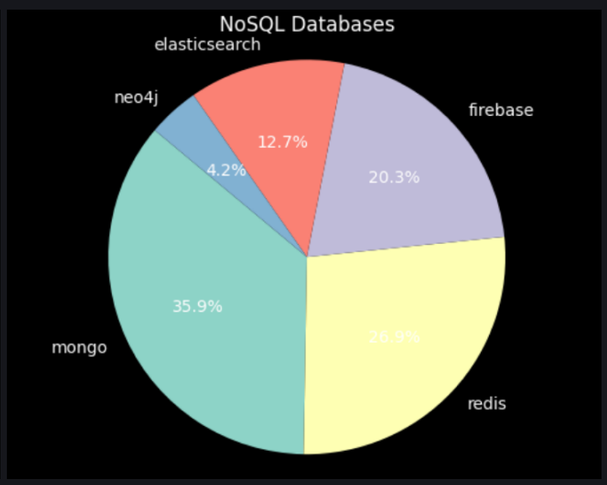
Set Expectations	Collect Data	Match Expectations & Data
The questions is interesting and answerable.	it is defintely will help developers to know what are the common bundle between frameworks and databases so learners can stick to the most common bundle.	Both are matching

Exploratory Data Analytics





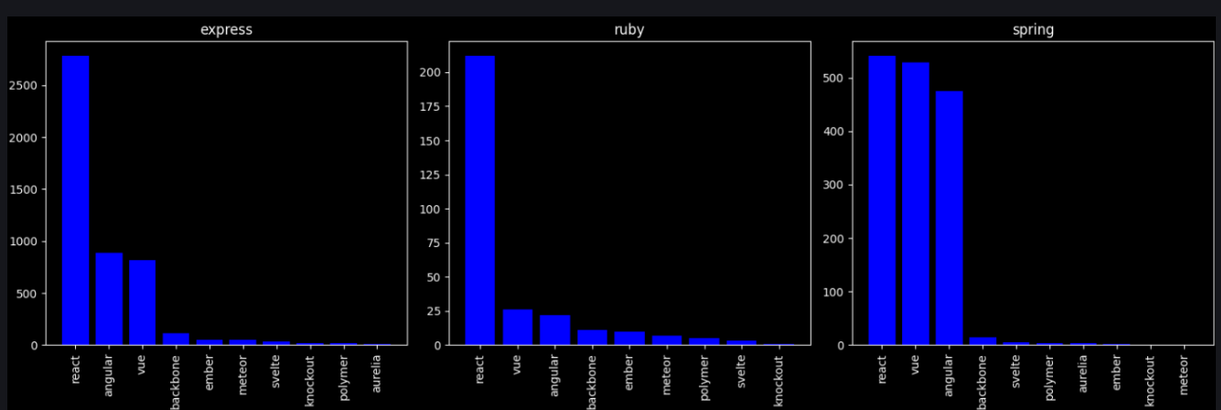




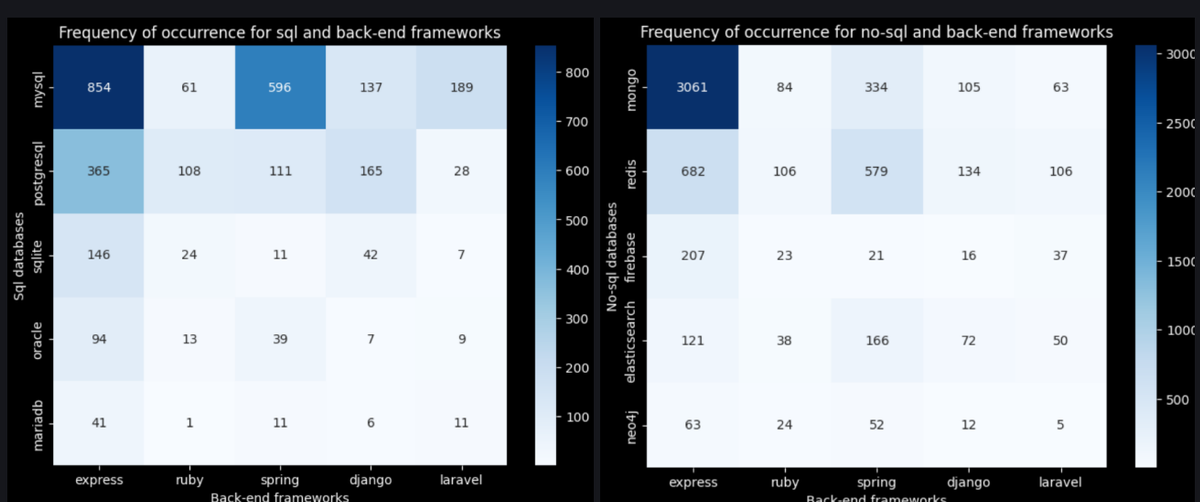
Set Expectations	Collect Data	Match Expectations & Data
I Expect that the exploratory data analysis will reveal the percentage of each framework/database and the dataset at hand covers variety of them	Indeed, the dataset covers a variety of frameworks and databases	Expectations are met

Model Building

✦ What front-end frameworks tend to occur most often with the three most frequent backend frameworks?



✦ What databases tend to occur more often with different backend frameworks?



Set Expectations	Collect Data	Match Expectations & Data
The model will be able to show the co-occurrence of frameworks and databases	Data is as shown	Of course, the model is able to show the co-occurrence of frameworks and databases demonstrating where the most common bundles

Result Interpretation

Front End Frameworks used with the top 3 back end frameworks

- The top 3 back end frameworks are: Express, Ruby, and Spring
- React is the most used front end framework no matter what back end framework is used
- Express + React is a the most common combination

Back End Frameworks used with SQL and NoSQL Databases

- The sample at hand shows that express is the most used back end framework
- Also it's easy that express is mostly used with NOSQL databases
- The matrix the shows the co-occurrence of backends and NOSQL is peaked at express and mongoDB
- The matrix the shows the co-occurrence of backends and SQL is more spread out

Set Expectations	Collect Data	Match Expectations & Data
I expect that React will dominate the frequency among other frameworks as it's powered by Meta. I also expect that Express will be the most common backend framework as it's a Node.js framework and Node.js is a runtime environment for JavaScript (The most popular programming language). Hence, I expect the most common bundle to be React-Express.	Data is as shown	Expectations are met perfectly

Set Expectations	Collect Data	Match Expectations & Data
I expect that Express will dominate the frequency among other frameworks as justified earlier. I expect Mongo and Express to be the most common bundle due to the seamless integration between the two.	Data is as shown	Expectations are met perfectly

Communicating Results

Front End Frameworks used with the top 3 back end frameworks

- There are high opportunties for developers who know React and Express
- The market is concentrated on a few back end frameworks like Express, Ruby, and Spring
- The market is concentrated on a few front end frameworks like React, Angular, and Vue
- Novoice developers interested in joining the market should stick to these common backend-frontend bundles

Back End Frameworks used with SQL and NoSQL Databases

- There are high opportunties for developers who know MongoDB and Express
- For beginners who want to learn a Nosql database, MongoDB seems a rational choice
- The variance of sql databases is greater than nosql databases

Set Expectations	Collect Data	Match Expectations & Data
Professor & TA will appreciate and be interested in the communicated information.	As above.	Will be known after the presentation day.

Language Associations

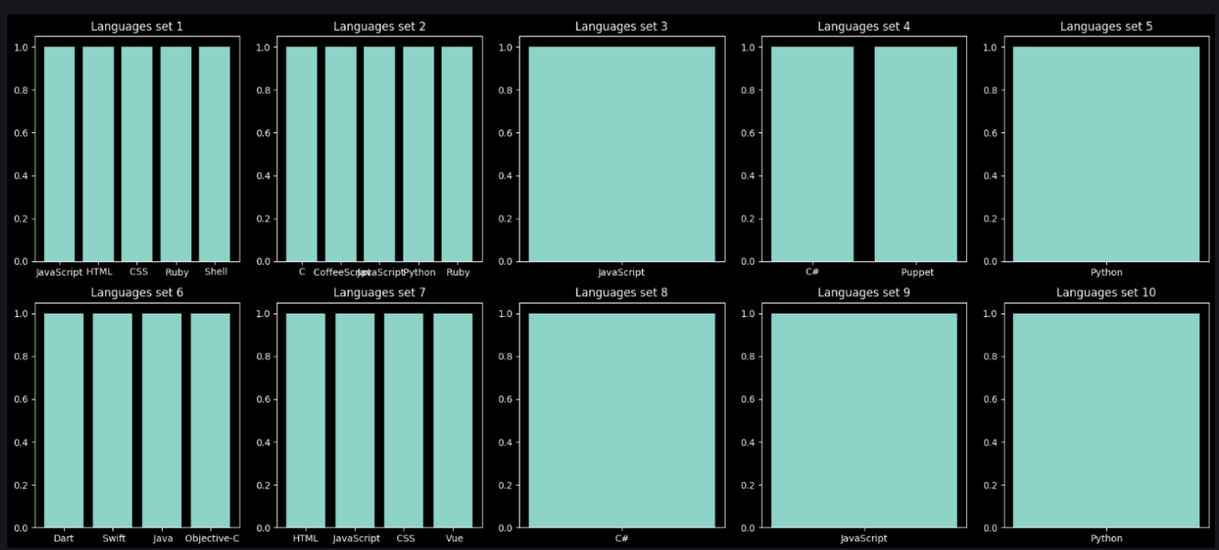
Stating Questions

✦ What are the top three sets of programming languages used together over time? What are some strong association rules that can be drawn from these sets?

Set Expectations	Collect Data	Match Expectations & Data
The question is interesting and answerable.	The question is already interesting as many software developers and software companies owners need to know its answer, and it is answerable as the data is available and with some statistical techniques we can answer it.	Expectations and data match.

Exploratory Data Analytics

Below is a sample plot depicting a list of lists, each containing a set of programming languages used together in a repository.



Set Expectations	Collect Data	Match Expectations & Data
There are many programming languages sets (that have more than one programming language) used together over time	As seen there more than 55 % of the used programming languages sets have more than one programming language.	Expectations and data match.

Model Building

The Apriori algorithm was used to solve this question. The first part of the question has been successfully answered by identifying sets with maximum support, which equates to finding the most common sets, the first part of the question has been successfully answered.

	support	itemsets
53	0.091709	(HTML, CSS)
71	0.067647	(HTML, JavaScript)
59	0.061143	(JavaScript, CSS)

The image shows the top three sets of languages used together and the support.

The second part of the question is solved by finding the rules with highest confidence.

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
1	(CSS)	(HTML)	0.144561	0.171556	0.091709	0.634396	3.697889	0.066909	2.265960	0.852867
0	(HTML)	(CSS)	0.171556	0.144561	0.091709	0.534571	3.697889	0.066909	1.837959	0.880658
7	(JavaScript)	(CSS)	0.128585	0.144561	0.061033	0.474651	3.283391	0.042445	1.628325	0.798055
9	(JavaScript)	(HTML)	0.128585	0.171556	0.059238	0.460690	2.685355	0.037178	1.536116	0.720219
3	(JavaScript)	(HTML)	0.155423	0.171556	0.067647	0.435246	2.537045	0.040983	1.466911	0.717330
5	(CSS)	(JavaScript)	0.144561	0.155423	0.061143	0.422953	2.721300	0.038674	1.463618	0.739420
6	(CSS)	(JavaScript)	0.144561	0.128585	0.061033	0.422195	3.283391	0.042445	1.508147	0.812959
2	(HTML)	(JavaScript)	0.171556	0.155423	0.067647	0.394315	2.537045	0.040983	1.394417	0.731300
4	(JavaScript)	(CSS)	0.155423	0.144561	0.061143	0.393394	2.721300	0.038674	1.410206	0.748929
8	(HTML)	(JavaScript)	0.171556	0.128585	0.059238	0.345297	2.685355	0.037178	1.331007	0.757577

The image shows the most important rules

Set Expectations	Collect Data	Match Expectations & Data
The questions can be answered using aprori algorithm.	Apriori algorithm is applied and the needed answers is obtained.	Expectations and data match.

Result Interpretation

Top most frequent languages and rules

- The Top first itemset is [' CSS', ' HTML'] with a support of 9.17%.
- The Top second itemset is ['JavaScript', ' HTML'] with a support of 6.76%.
- The Top third itemset is ['JavaScript', ' CSS'] with a support of 6.11%.
- The Top first rule is CSS -> HTML with a confidence of 63.44%.
- The Top second rule is HTML -> CSS with a confidence of 53.46%.
- The Top third rule is JavaScript -> CSS with a confidence of 47.47%.
- The Top fourth rule is JavaScript -> HTML with a confidence of 46.07%.

Set Expectations	Collect Data	Match Expectations & Data
Web development programming languages are the most used programming languages sets.	HTML, CSS, and JavaScript are the most appeared in the resulted most used programming languages sets.	Expectations and data match.

Communicating Results

Top most frequent languages and rules

✦ As In web programming many languages are used together, the most frequent itemsets are of size are all for web programming languages.

✦ Also the top rules are between web programming languages.

✦ The Top first itemset is [' CSS', ' HTML'] with a support of 9.17%.

✦ The Top second itemset is ['JavaScript', ' HTML'] with a support of 6.76%.

✦ The Top third itemset is ['JavaScript', ' CSS'] with a support of 6.11%.

✦ The Top first rule is CSS -> HTML with a confidence of 63.44%.

✦ The Top second rule is HTML -> CSS with a confidence of 53.46%.

✦ The Top third rule is JavaScript -> CSS with a confidence of 47.47%.

✦ The Top fourth rule is JavaScript -> HTML with a confidence of 46.07%.

Set Expectations	Collect Data	Match Expectations & Data
The results are logical and the professor will be satisfied.	The shown results.	Will see soon.

Licenses, Languages & Sizes

Stating Questions

✦ What licenses are associated with what primary programming languages and project sizes?

Set Expectations	Collect Data	Match Expectations & Data
The question is interesting and anwerable	The question is inertesting as it is unknown and answerable as the data is available so by some statistical analysis we can find the answer.	Expectations and data match.

Exploratory Data Analytics

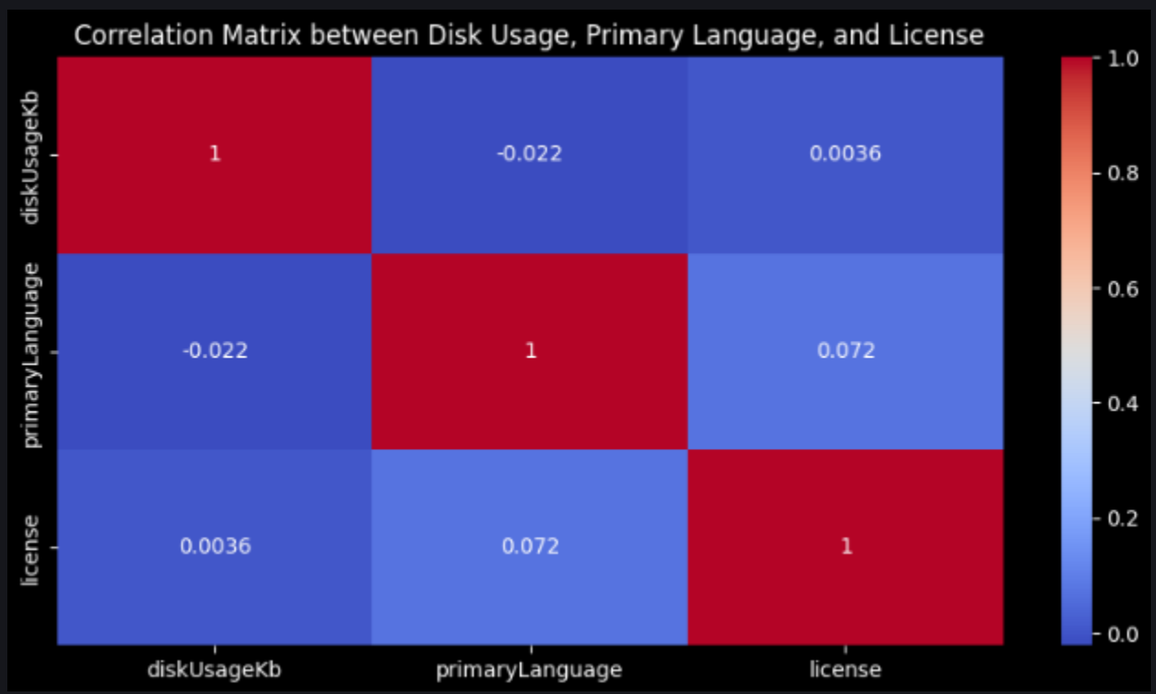
After exploring the data, the following findings have been obtained:

```
Number of unique languages:  377
Number of unique licenses:  44
Max project size: 41796529 KB
Min project size: 0 KB
Standard deviation of project sizes: 269947.31862926163 KB
```

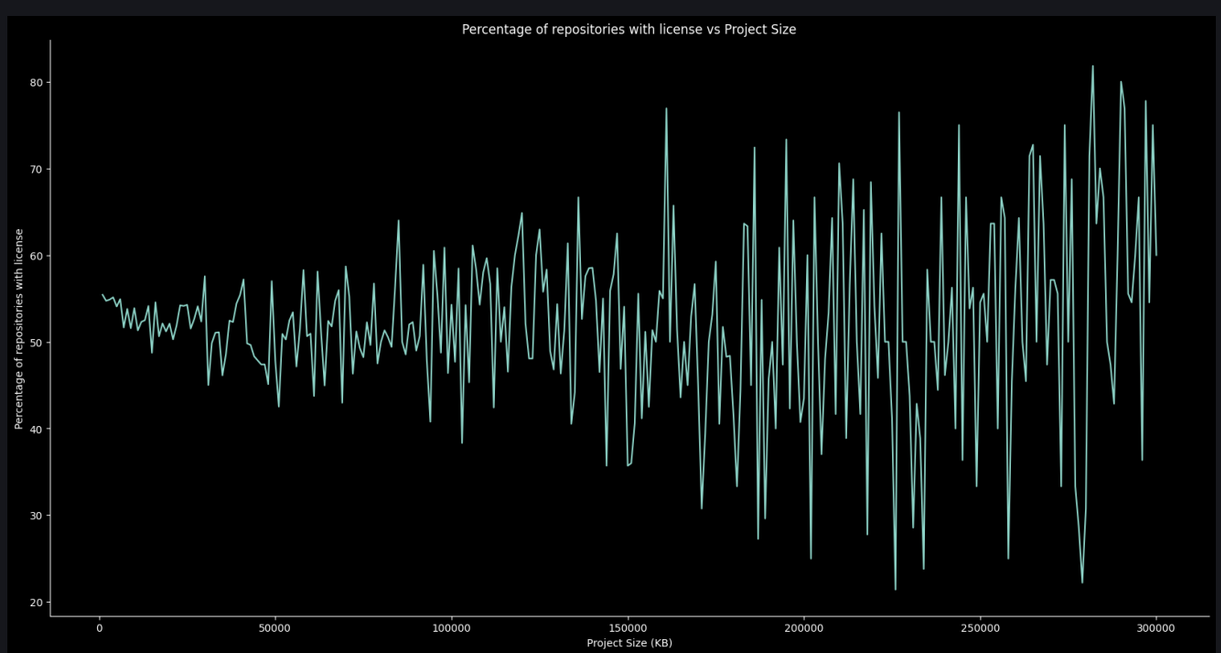
Set Expectations	Collect Data	Match Expectations & Data
There is many used languages on github, and many licenses, and many project sizes.	There is 377 languages, 44 licenses and the standard deviation of the size of projects is about 270 Megabytes.	Expectations and data match.

Model Building

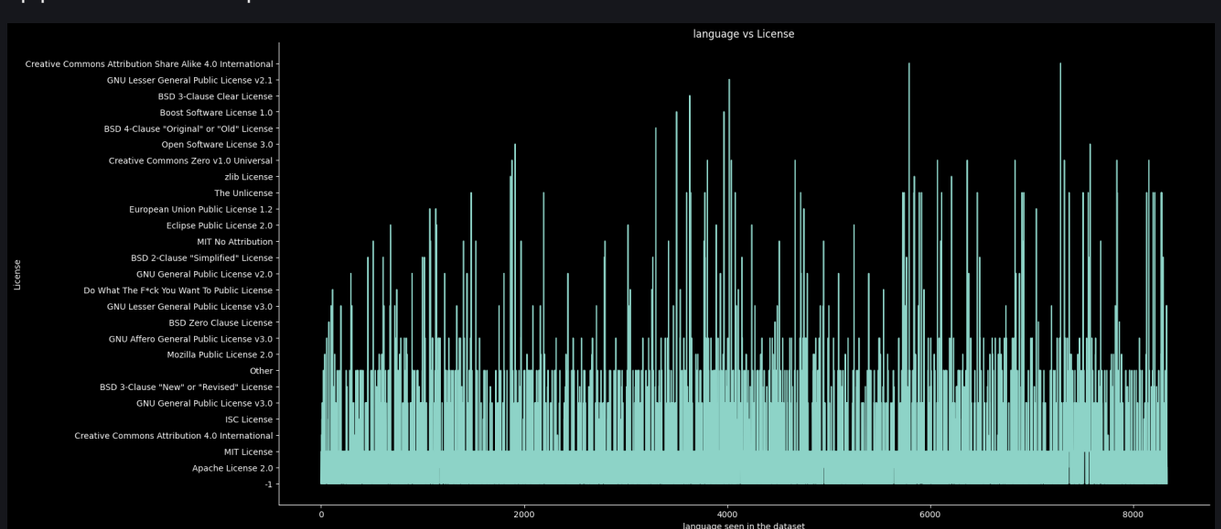
First the correlation matrix between the disk usage, licenses, and primary languages was calculated and plotted



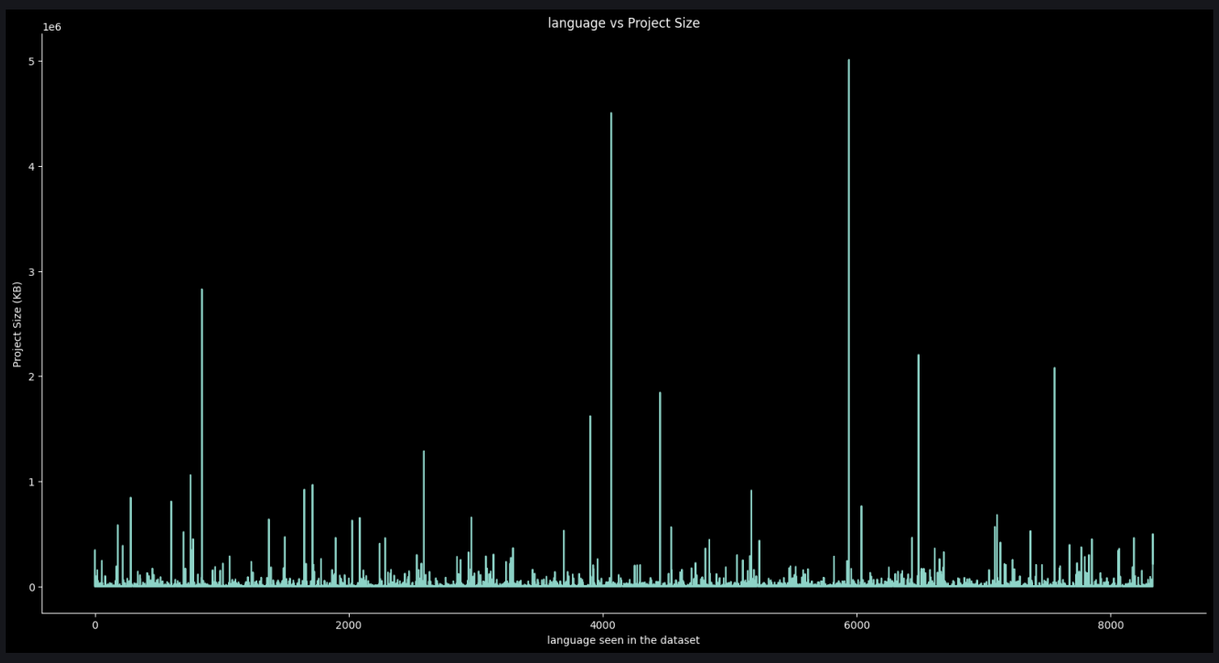
The percentage of repositories that have licenses for each project size is calculated and plotted



For many language samples the license used for the language in the language appearances are plotted



For many language samples the project size for this language appearances are plotted



Set Expectations	Collect Data	Match Expectations & Data
Some statistical analysis on the data wiil be completely enough to answer the question.	After Ploting the relation between each pair we can prove or disprove the relation between them.	Expectations and data match.

Result Interpretation

Relationship between licenses, languages and project size

- ✦ There is no seen relationship between project size and license
- ✦ There is no seen relationship between project size and language
- ✦ There is no seen relationship between license and language
- ✦ So there is no seen relationship between license, language and project size

Set Expectations	Collect Data	Match Expectations & Data
No strong relation between languages and licenses and project sizes.	As seen in the plots the size changes randomly with the change of language or license, and also languages and licenses are not related.	Expectations and data match.

Communicating Results

Relationship between licenses, languages and project size

✦ Any language can be used to create large and small projects so there is no seen relationship between project size and language

✦ Choosing license depends more on the code owner not the used language or the project size, as same user may use different programming languages and create projects with different sizes

✦ So there is no seen relationship between license, language and project size

Set Expectations	Collect Data	Match Expectations & Data
the doctor will be satisfied with the results as the results are clear and easy to understand and logical.	The Written above results.	Will know soon.

Generalizing Archival Trends

Stating Questions

✦ Does the language with the most archived repositories in the period (2009 and 2015) generalize to the whole period (2009 to 2023) ?

Set Expectations	Collect Data	Match Expectations & Data
The question is specific, interest and novel.	The question is specific and answerable using a statistical test. It is interesting as it will help the business in the chioce of the programming language to use in the next project (the question is directly related to the community support). It is novel as my research did not find any similar question.	The question is indeed specific, interest and novel.

Exploratory Data Analytics

Two tables that contains the archival rates for the 2 periods (2009-2015) and (2009-2023)

2009-2015

	primaryLanguage	sum	count	archival_rate
0	Ruby	362	3360	0.107738
1	PHP	378	3617	0.104506
2	JavaScript	924	10417	0.088701
3	Python	491	6427	0.076396
4	Java	358	4751	0.075353
5	Document	240	3227	0.074372
6	C	149	3065	0.048613
7	C++	142	3007	0.047223

2009-2023

	primaryLanguage	sum	count	archival_rate
0	Ruby	444	4732	0.093829
1	PHP	660	8246	0.080039
2	JavaScript	2116	32684	0.064741
3	CSS	199	3157	0.063035
4	Shell	362	6021	0.060123
5	Go	403	6723	0.059943
6	Rust	168	2895	0.058031
7	C#	433	7820	0.055371

With ruby to be the top archival rate in both (2009-2015) and (2009-2023)

Set Expectations	Collect Data	Match Expectations & Data
There are ratio of archived repositories is low across all languages.	The data is shown in the above table.	The data matches the expectations.

Model Building

Two proportions z-test

Proportion difference test results

✦ The difference between 'Ruby' and the other languages is statistically significant

Set Expectations	Collect Data	Match Expectations & Data
The test is appropriate to answer the question.	The test was applied to the data and the results are shown in the above cell.	The model was able to answer the question.

Result Interpretation

✦ Ruby has the highest archival rate with statistical significance ($p < 0.01$) between it and all the languages

Set Expectations	Collect Data	Match Expectations & Data
The analysis will answer the question.	The answer is shown in the above cell.	The question was answered.

Communicating Results

Insights

✦ All languages have an archival rate of around 10% or less

✦ Ruby has the highest archival rate with statistical significance ($p < 0.01$) between it and all the languages

Set Expectations	Collect Data	Match Expectations & Data
Professor & TA will appreciate and be interested in the communicated information.	As above.	Will be known after the presentation day.

Generalizing Dynamically Typed Trends

Stating Questions

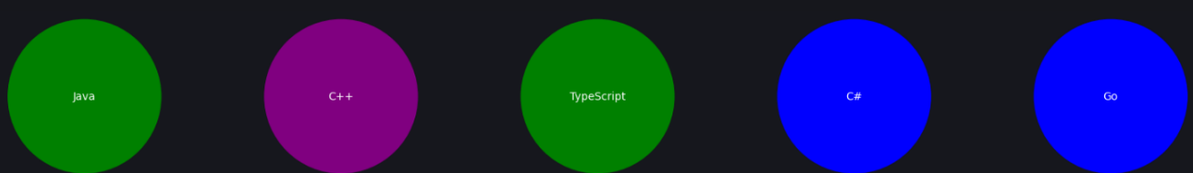
✦ Does the popularity of Javascript generalize to prove that dynamically typed languages are more popular than statically typed ones?

Set Expectations	Collect Data	Match Expectations & Data
The question discusses interesting phenomena and tries to reveal if really dynamically typed languages are more popular than statically typed ones based on the popularity of Javascript. I expect that there is no explict answer for such question.	After a little searching over internet, there is not explict answer for such question	Expectations and data are matched

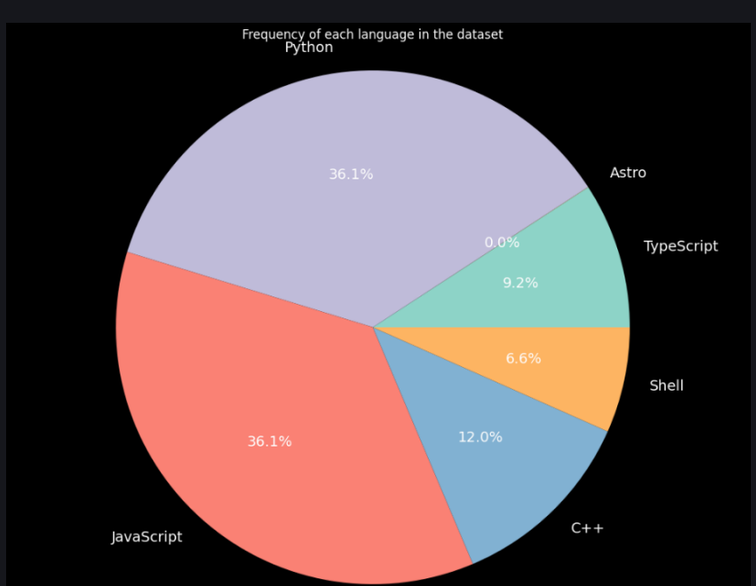
Exploratory Data Analytics

Top 5 dynamically typed programming languages

Top 5 statically typed programming languages

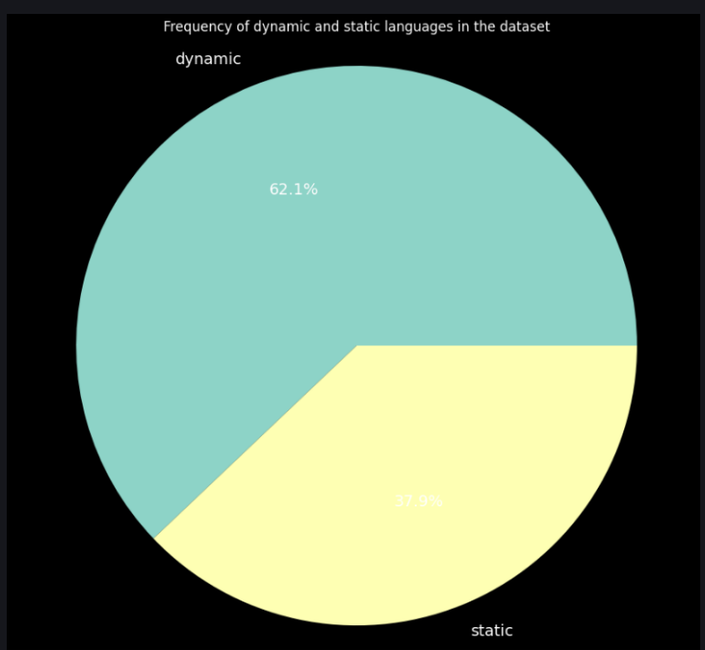


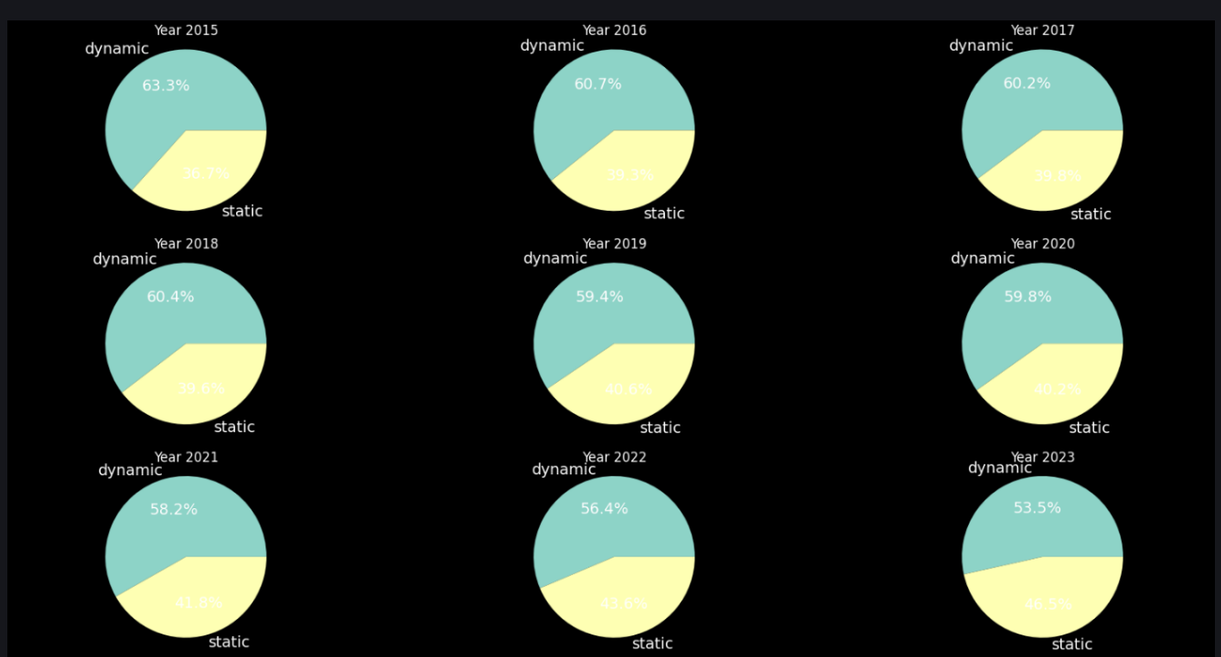
JavaScript is really a popular programming language?



Set Expectations	Collect Data	Match Expectations & Data
I expect JavaScript to be the most popular programming language as stated in the question I expect that dynamically typed languages and statically typed language are represented in the dataset adequately.	After doing a descriptive analysis the data is as shown above.	Expectations and data are matched

Model Building





Hypothesis Testing

Hypothesis: The average usage of dynamically typed programming languages is greater than the average usage of statically typed ones

Proportion Z-Test:

by evaluating the total number of repositories each type has and applying the

test relying on statsmodels library with significance level = 0.05 where:

$H_o : P_d <= P_S$

$H_1 : P_d > P_S$

The setup mentioned above yielded in rejection for the null hypothesis and accepting the alternative hypothesis.

Set Expectations	Collect Data	Match Expectations & Data
I expect that the model will be able to present the usage of each type of programming language.	Data is as shown above	Yes, the model is able to present the usage of each type of programming language in a pie chart.

Result Interpretation

Dynamic vs Static Languages

✦ Dynamic languages are more popular than static languages and the reason is that they are easier to use and because they appeared as an super cool innovation which prevent the hassle the developers faced at the eariler days of programming specially in the 90s

✦ Static typing is increasing in popularity over the years as the world realized the dark side of dynamic languages which is the lack of type checking which leads to many bugs and errors so they started to use more clean and neat languages like Java and C# which are definitely easier than C++ and C

Set Expectations	Collect Data	Match Expectations & Data
I expected that results are interpretable and there are resonable explanations for the results.	Here are the results and explanations.	Indeed, the interpretations are logical and reasonable.

Communicating Results

Dynamic Languages vs Static Languages

✦ Over history, dynamic languages are more used than static languages

✦ The usage of dynamic languages is increasing over time

✦ The world is not biased towards dynamic languages in absolute terms, in fact the popularity of both types is changing over time due to external factors like the requirements of the market and the needs of the developers

Set Expectations	Collect Data	Match Expectations & Data
Professor & TA will appreciate and be interested in the communicated information.	As above.	Will be known after the presentation day.

Expected Language Archivals

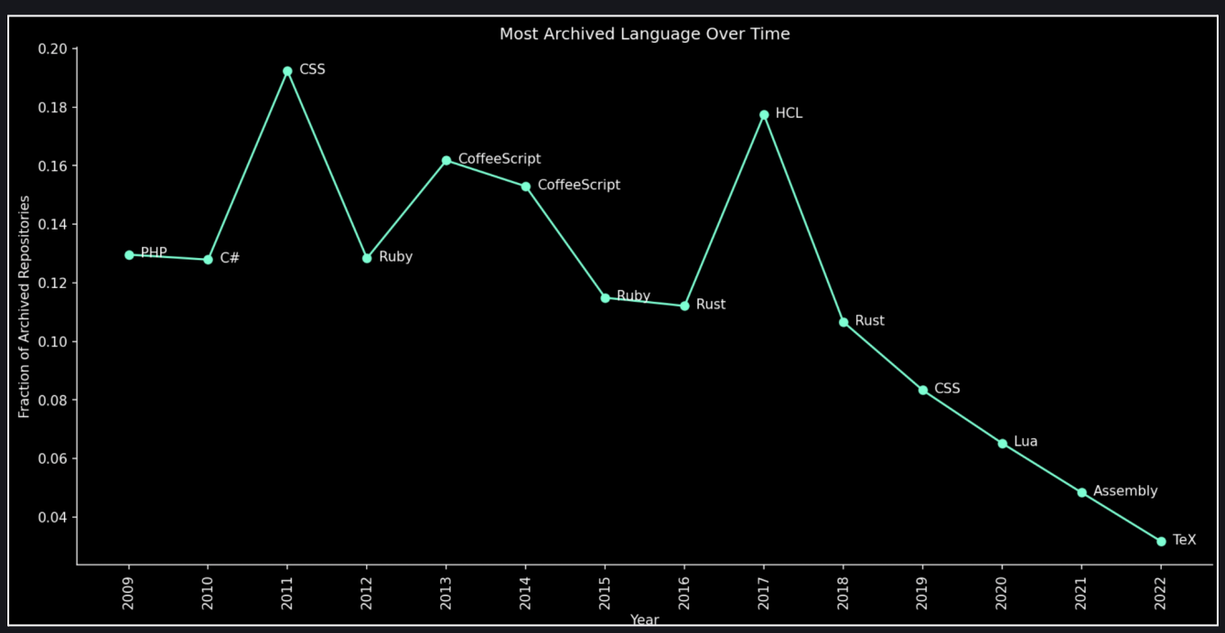
Stating Questions

✦ What programming language is expected to have the most repos archived in 2023?

Set Expectations	Collect Data	Match Expectations & Data
Question is interesting, specific, answerable and novel.	Question is interesting as this would illustrate which language can we declare as the oldest for 2023. It is specific as is obvious and its answerable using time series analysis. Besides, its novel as this is not a known fact although expectations can be made.	Data and expectations match.

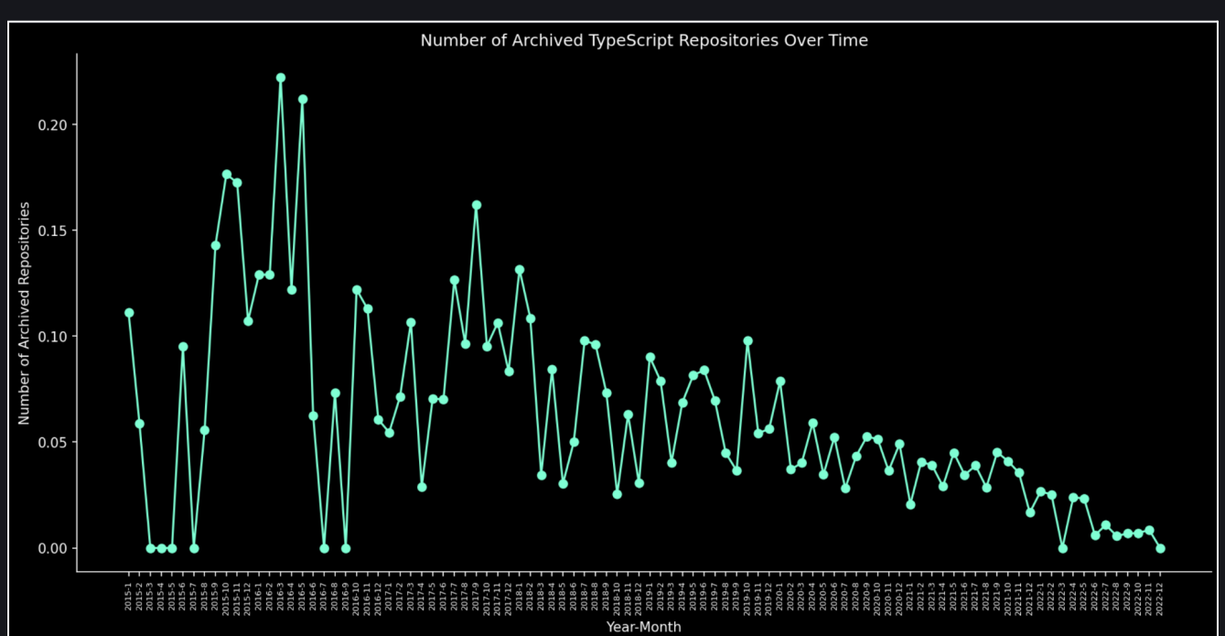
Exploratory Data Analytics

Let's find the most archived language every year and plot it



Set Expectations	Collect Data	Match Expectations & Data
We expect to see old languages like Assembly in the list	Data is as shown.	They didn't match until it was realized that niche languages (e.g., CoffeeScript, HCL) can appear and die quickly without being heard of

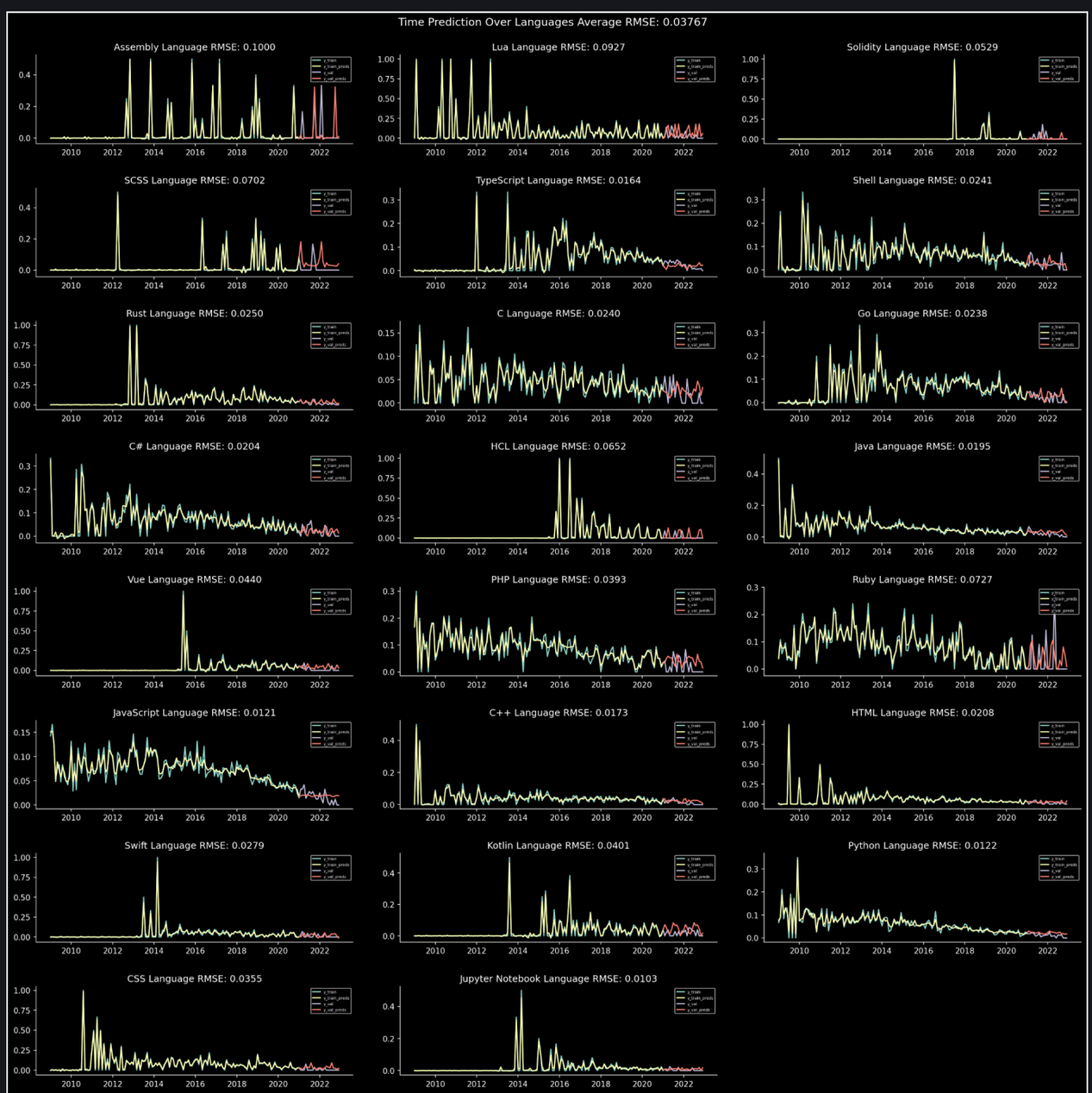
Let's consider the monthly archivals for C over the last 8 years (2015-2022)



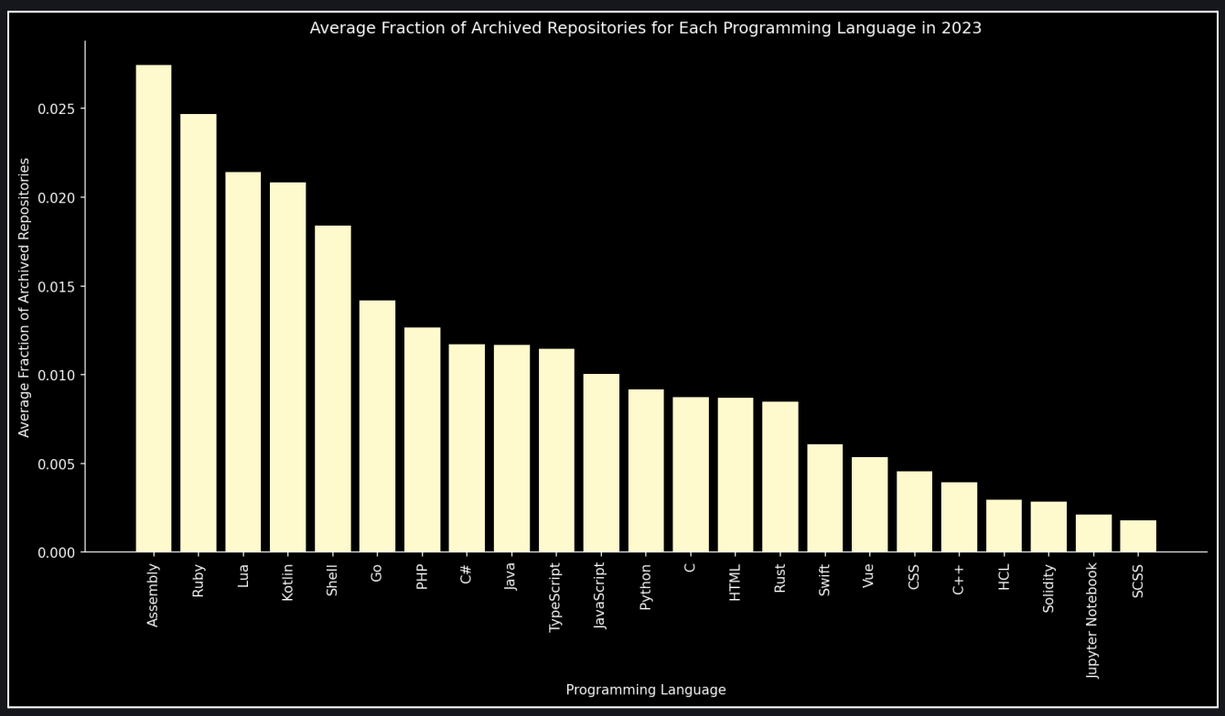
Model Building

Let’s build a model for time series forecasting. We will use XGBoost and we will do this for each language over the set of programming languages that satisfy having at least 50 repos in 2022 with at least three archivals.

The results from the model are as follows for that set of languages



Now let’s predict for each month in 2023 then average and sort the results to get



Result Interpretation

Insights

✦ Assembly is expected to be the most archived language in 2023. It makes sense as its one of the oldest languages around

✦ Different languages seem to follow trends of different complexity. Trend is mostly decreasing for modern popular languages but stochastic for older ones.

✦ C++ and C seem to be safer than expected; which can be justified by their use in embedded systems, OSs and libraries for other languages

✦ Niche languages like HCL and Solidity don't seem to be at risk but they probably took a big hit earlier

✦ The endangerment of languages like Ruby and Lua is expected. Lua has been recently listed in a popular list of the worst languages and Ruby stopped shining after the rise of Python and JS

Set Expectations	Collect Data	Match Expectations & Data
Most archived language in 2023 isn't expected to be a modern language but rather an old or low-level one	Data is as shown	Indeed, Assembly fits such position

Communicating Results

Insights

✦ Assembly is not dead yet but it seems to be in the process.

✦ It's not time yet for C and C++ to die. High performance replacements either don't exist or are not popular enough.

✦ Ruby and Lua are also at risk of endangerment.

Expected Python Contributions

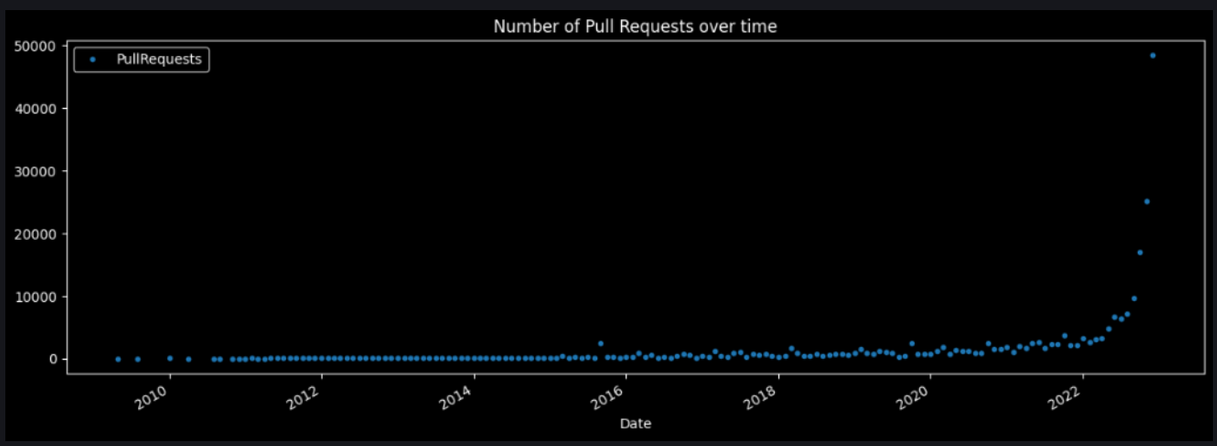
Stating Questions

✦ What is the expected number of pull requests over all Python repositories that were made in 2023?

Set Expectations	Collect Data	Match Expectations & Data
The question is interesting and answerable from the data.	The question is interesting because software companies need to predict python contributions for 2023, and it it is answerable because we have data on repositories languages, creation dates, and pull requests.	Expectations and data match.

Exploratory Data Analytics

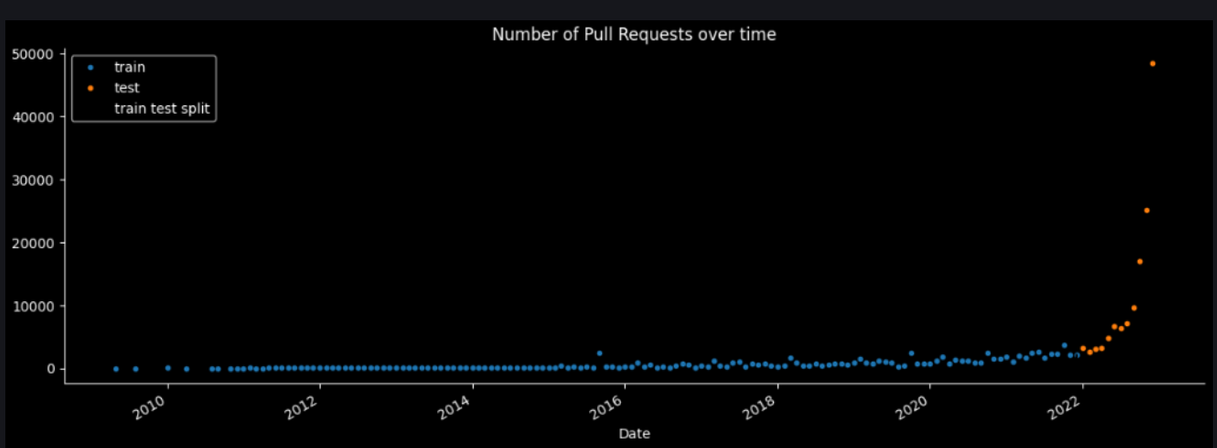
Number of pull requests for python repositories created at each month from January 2009 to December 2022 are calculated and ploted



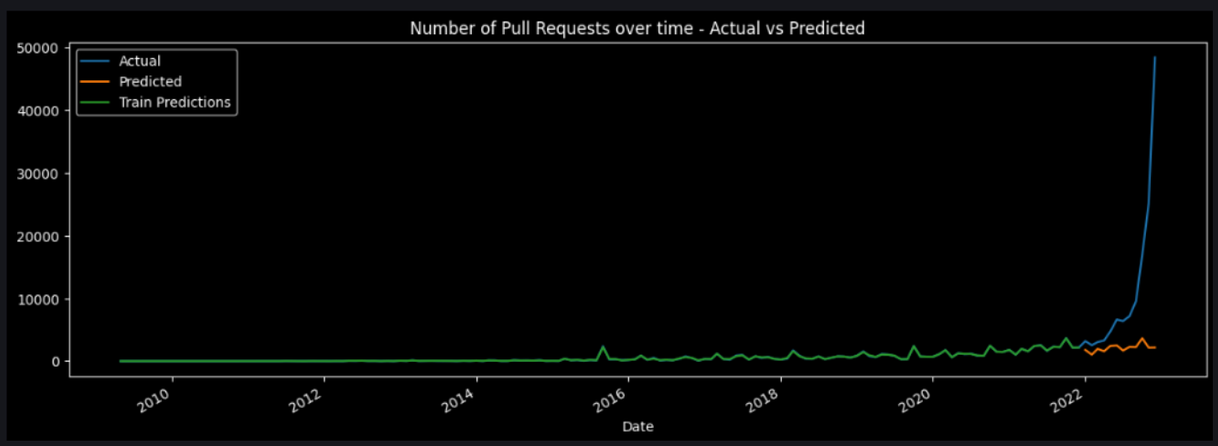
Set Expectations	Collect Data	Match Expectations & Data
There is more than 25000 python repositories on the data and theses repositories have over 100000 pull requests.	Already we have 29693 python repositories and 214164 pull requests.	Expectations and data match.

Model Building

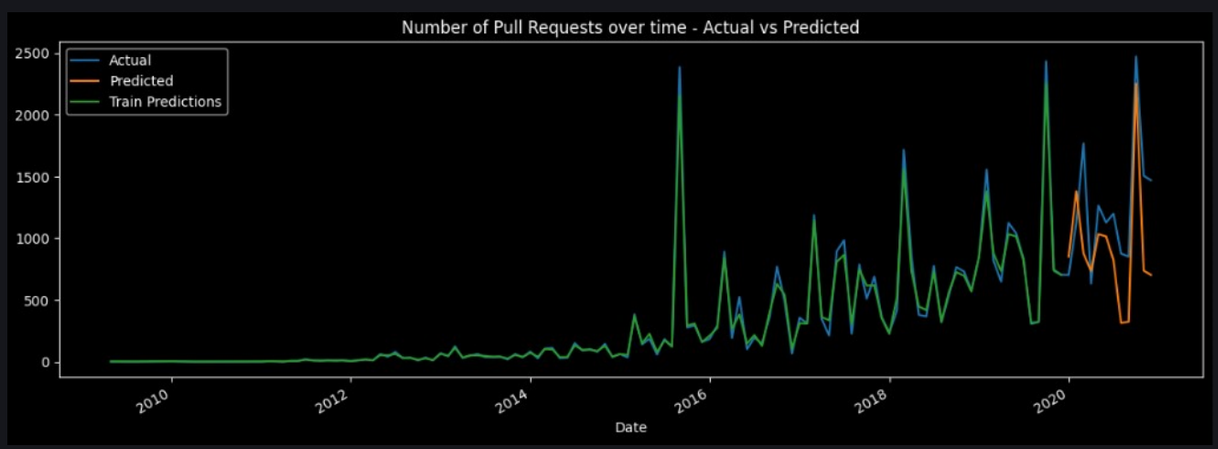
The data has been divided into two regions: the training region, encompassing data up until 2022, and the testing region, consisting of data from after 2022.



XGBoost, a regression algorithm, was employed to predict the number of pull requests in 2022. However, the performance of the model did not meet the desired level of acceptability to be used for predicting pull requests in 2023.



The reason for the unacceptable performance of the XGBoost regression model in predicting pull requests for 2022 could be attributed to the significant dissimilarity observed in the 2022 pull request data compared to the previous data. As the same model is used to predict 2021 pull requests given the data from 2009 to 2020 and gives good results.



Multiple programming languages were analyzed in an attempt to predict the number of pull requests for 2022. Surprisingly, it was discovered that all languages exhibited distinct patterns in the 2022 pull request numbers compared to previous years. This indicates the possibility of changes made by GitHub to their pull requests API, as this anomaly was not limited to Python but affected all languages.

Set Expectations	Collect Data	Match Expectations & Data
The difference in 2022 data may lead to that the question is hard to answer using the model.	Already the question is hard to answer using regression models	Expectations and data match.

Result Interpretation

Insights about expecting python contributions

✦ The model is able to predict before 2022 with a good accuracy

✦ Due to the difference in the number of pull requests between 2022 and rest of the years, the model is not able to predict the number of pull requests in 2022

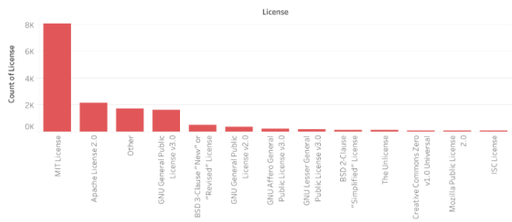
✦ So the model cannot be used to predict the number of pull requests in 2022

Communicating Results

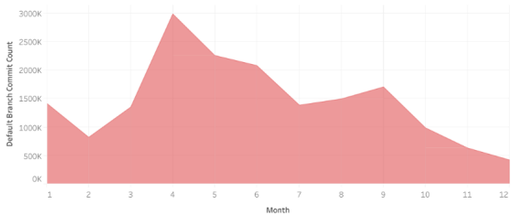
Same as results

Dashboards

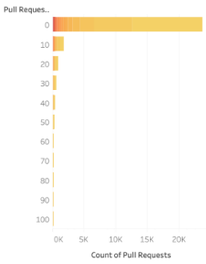
Common Licenses



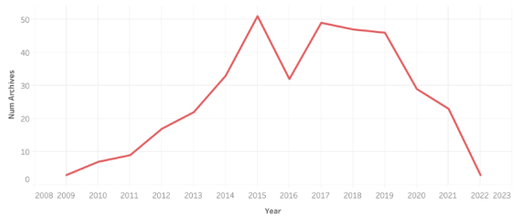
Average Commits Per Month Since 2009



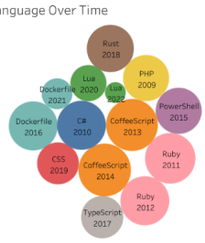
Number of Pull Requests & Stars



C++ Archivals Over Time



Most Archived Language Over Time



License or Not