Analyzing Trends in Open Source Software Contributions

Andrew Casner
University of Colorado, Boulder
andrew.casner@colorado.edu

Oliver Collins
University of Colorado, Boulder
oliver.collins@colorado.edu

Carl Cortright
University of Colorado, Boulder
carl.cortright@colorado.edu

Shubha Swamy
University of Colorado, Boulder
shubha.swamy@colorado.edu

1. Abstract

Over time, the most popular programming languages have shifted dramatically. And while there might be an overall trend with the rise and fall of these languages, often it can be quite difficult to predict these fluctuations. So for this project, we have decided to dive into a large open-source dataset that looks into contributions to open-source software and try to map what we hope, will be an accurate representation of these trends. Acquired from libraries.io, this dataset with over 397+ million rows provides us with repository names, timestamps, programming languages, and many more attributes related to package managers supported through the platform. Providing this information gave us a lot of information, yet required a lot of cleaning, analyzing, and parsing.

1.1 Important Questions

The primary questions we will consider during the duration of this project will focus on analyzing trends in open source projects. What insights can we gain to improve the open source community further? How can we identify areas in the open source community that need improvement? And can we predict upcoming popular repositories?

1.2 Brief Summary

We based a lot of our analysis on trends we thought would exist between attributes.

2. Introduction

Our primary goal of this project is to gain insights about open source software. We will accomplish this goal by mining data from a dataset about open source projects. We will focus our work on four main areas: tracking trends in programming languages, analyzing how popular repositories have changed over time, how contributions to those repositories have changed over time, and also monitoring repository life cycles.

2.1 Questions

We will consider a few central questions to help guide us through the data mining process to gain better intuition about open source software. The first question we seek to answer is: “What insights can we gain to improve the open source community further?”. As we use various techniques to mine the data throughout the project, we hope to understand more about the information various languages used, contributions to popular repositories, and trends in dependencies which rely solely on those open source projects. The second question we hope to answer is: “How can we identify areas in the open source community that need improvement?”. This question will not only help us understand the missing aspects of the open source community but hopefully bridge a gap within the community. To find answers to this question, we will analyze trends in contributions and repositories to figure out areas of improvement within the open source community. And finally, we aim to answer the most driving question of our project: “Can we predict upcoming popular repositories?”. We hope

to answer this question using all the information from the mined data. Our team would like to predict upcoming popular repositories based on all the insights we gain from the data analysis we will perform. Answering this question will be beneficial to understanding the open source community.

3. Related Work

3.1 Literature Survey

There has been a significant body of research done on patterns in open source contributions. Every year Stack Overflow, a popular site for asking and answering coding questions, does a poll of the developers on their platform [1]. This research is used to track popular languages and libraries developers are using on the platform. Other research focuses on the potential impacts of research on open source software on the broader computer science community [2]. This research can help act as a guide as we decide what areas of open source development are important to focus on in our data mining project. The majority of the research in this area has focused specifically on Github repositories, attempting to determine the influence of any given project [3]. In our work, with the dataset we have access to, we will attempt to extend this research to all open source projects independent of where they are hosted or built. Other research by the Apache foundation has shown that the majority of open source contributions are made in a way that is not collaborative but instead driven by individual developers [4].

4. Dataset

The dataset we will be using for our project comes from a company called libraries.io. The purpose of libraries.io is to monitor open source projects to help developers better understand dependencies for their projects.

The dataset itself contains 311 million data points from 34 package managers and three source code repositories, including npm, GitHub, PyPi, RubyGems, Maven, Bower, and other large, language-specific package managers. With this breadth, libraries.io can track over 2.7 million unique open source packages, spanning 31 million repositories, tracking 161 million dependencies between them [5]. With this large of a dataset, we expect to have many different variables of interest that we can mine.

The dataset comes packaged in several large CSV formatted files. The main file that we will be mining is the projects database. This file contains all of the individual projects libraries.io is tracking. This file also has vital attributes for each project including language, status (active, depreciated, etc.), dependent projects count and more. Other important datasets include the dependencies CSV, which has detailed information on interdependencies between projects. Using this module we can build a dependencies graph, and mine information about which projects are most important in the open source ecosystem.

5. Main Techniques Applied

5. 1 Data Cleaning and Preprocessing

Data cleaning and preprocessing is a vital component of our project. It is important that we work with a dataset with good quality data in order to gain accurate results. We will conduct a series of important steps in order to ensure we use data with a good standard. Our data preprocessing will focus on five main areas including data cleaning, integration, transformation, reduction, and discretization. These steps will ensure that our final results we will gain from our analysis will be as accurate as possible.

5. 1.1 Proposed Work

Since this dataset is already in excellent condition, not much will have to be performed on the dataset. However, some necessary data cleansing will need to be completed such as scrubbing the dataset to remove null values and synchronizing time zones. Once these simple tasks are complete, our team will perform more advanced data preprocessing techniques such as matching a unique user across multiple package managers.

Once our team has appropriately cleaned our data, we will then decide which patterns answer the questions asked of the dataset. Those will be used to create visualizations, such as bar plots and graphs to show the trends found in the given data. Using the data shifted over time. To analyze how popular repositories have changed over time, we will investigate how project dependencies of various open source projects have changed over time. To analyze how contributions to repositories have changed over

	Unnamed: 0	Fork	Created Timestamp	Updated Timestamp	Last pushed Timestamp	Homepage URL	Size
0	1	2014-09-15 01:21:34 UTC	2016-12-28 16:33:17 UTC	2016-12-18 18:31:32 UTC	http://brianmhunt.github.io/knockout-modal/	512	7
1	2	2010-11-01 09:27:43 UTC	2018-02-11 10:04:55 UTC	2017-06-21 22:54:45 UTC	NaN	924	543
2	3	2014-09-13 03:14:07 UTC	2017-03-18 22:40:02 UTC	2015-01-14 02:01:03 UTC	NaN	472	1
3	4	2014-12-27 21:02:09 UTC	2016-12-28 16:45:20 UTC	2015-01-07 18:04:42 UTC	http://zonuexe.github.io/aozora-ruby-parser.js/	536	3
4	5	2014-12-04 21:13:48 UTC	2017-03-18 22:40:04 UTC	2014-12-11 16:12:08 UTC	http://rawgit.com/immense/knockout-pickatime/m...	192	1
5	6	2014-12-04 17:04:45 UTC	2016-11-03 09:12:51 UTC	2015-10-16 18:13:50 UTC	http://rawgit.com/immense/knockout-pickadate/m...	285	3
6	7	2014-11-24 22:08:11 UTC	2017-08-07 19:18:03 UTC	2016-06-20 20:09:51 UTC	NaN	771	0

available, we will construct a dependency graph of all of the dependencies in the entire open source ecosystem. We will then conduct an eigenvector analysis on this graph to find the most influential projects. This same analysis can then be done on language-specific sub-graphs to find the most influential projects in each language.

Finally, we will create a write up of what we have learned and possibly publish it online in a way that is consistent with the open source mentality that we are studying. It is essential to our team that what we learn about the open source community can be freely and openly shared. Our write up is also aligned with the license that the libraries.io dataset is under.

5.1.2 Evaluation Methods

We will be gathering various conclusions from our dataset. Our project focuses on four areas: tracking trends in programming languages, analyzing how popular repositories have changed over time, how contributions to those repositories have changed over time, and also tracking repository life cycles. We will need to utilize various evaluation methods to draw insights about those topics. To track trends in programming languages over time, we will analyze the frequency of popular programming languages used in various open source projects and how that has time, we will first narrow down our data range to top contributed repositories, and then we will analyze the rate at which the contributions to those repositories have changed over time. Through this, we will also be able to conclude information about repository life cycles. By looking at trends and dependencies of various repositories over time, we will be able to conclude the repository's life cycle.

In addition to analyzing our data in regards to the four main areas we will be focusing on, we will also compare our results to conclusions found in the Literature Survey section (2).

5.1.3 Tools

We will be using several tools to mine and analyze the dataset to reach our conclusions about open source software. The primary programming language in this project will be Python. We will be using various Python libraries to analyze, parse, and present our data. One of the tools we aim to use, Pandas, will be used for most of our data analysis. As for our computational and statistical analysis, we will be using SciPy and NumPy. Since our team will be presenting our findings through data visualizations, we will be using a plethora of different libraries including Matplotlib, Bokeh, graph-tool, Seaborn, and several others. Moreover, lastly, for a lot of our

numerical simulations, statistical modeling, data visualization, as well as a way to keep our code clean and organized we will be encapsulating our code in Jupyter Notebooks.

In addition, we will also use AWS Cloud 9 as a collaborative working environment. The Cloud 9 is a collaborative IDE that allows us to all contribute to the code base and run intensive data mining tasks on a much more powerful cloud computer.

5.1.4 Milestones

We will use the milestones outlined below in order to guide us through the process of mining our data. We will be watching the same timeline as outlined in the project description in regards to our milestones but we will mostly be following the milestone table our team has devised below for additional deadlines.

No.	Milestone	Due Date
1	Proposal Presentation*	February 27th
2	Proposal Paper*	March 6th
3	Data Hosting on AWS	March 20th
4	Data Cleaning & Preprocessing	April 1st
5	Create Test Data	April 7th
6	Progress Report*	April 10th
7	Data Visualizations	April 17th
8	Data Analysis	April 17th
9	Analyzing Results	April 20th
10	Application	April 20th
11	Interactive Site with Visualizations	April 23rd
12	Final Presentation*	April 23rd
13	Final Paper*	May 1
* indicates milestones (due dates) set by professor		

5.1.5 Data Hosting on AWS

To collaboratively work on this project, we used a shared platform to work collaboratively on this project. We settled on using Amazon Web Services Cloud 9 as our IDE, cloud computing resource, and storage solution for our project. A significant amount of time involved figuring out and setting up the proper EC2 Instance type and mounting an EBS volume to the Instance. Once we finally loaded and unzipped the dataset to the volume on AWS, we began cleaning and preprocessing the data.

5.1.6 Data Cleaning and Preprocessing

We parsed and cleaned the data to ensure the quality of the dataset even though it came pre-processed. We performed several processes to further enhance the quality of our dataset including data cleaning, data integration, data transformation, and data reduction. The primary purpose of these pre-processing methods was to be confident that the dataset was clean of incomplete, noisy, and inconsistent data. The quality of data is significant when it comes to concluding the information because higher quality data yields more accurate results [6].

We used several libraries in python to help us organize and clean the data. The primary library that we used to accomplish this was Python Pandas. Pandas is a python library used for data manipulation and analysis.Since our data set consisted of millions of data points, we used the head function is Pandas to help us preview the data. Previewing various parts of our dataset allowed us to not only understand the organization of our dataset better but also helped us determine what steps we needed to take to ensure a high-quality dataset. Using the describe function in pandas gave us a better insight into our data set. We were able to gain quick summaries of our overall dataset using the describe function in pandas for

various columns. For example, we were able to conclude the average dependencies for open source projects. This conclusion was crucial for us to recognize the significant trends in our dataset. We used other functions in Pandas to delve deeper into

```
# Get the data in form YYYY-MM-DD
dates = []
for x in range(10000):
    dates.append(a[x][0:10])
```

our data. Once we were able to find unusual patterns and had a greater understanding of our data, we continued through the cleaning process. By analyzing the count for each attribute, we concluded that individual data values were missing. Missing values contribute significantly to data quality problems. It is imperative to handle these missing values to ensure our data analysis will yield high-quality results. To fix this issue, we filled in numerical data with the attribute mean and categorical data with a global constant relating to its corresponding attribute. The data comes in six different packages-- projects, versions, tags, dependencies, repositories, repository dependencies, and projects with related repository fields. Since we will be concluding all these six packages, we had to ensure that we applied this cleaning process on all six packages.

After we completed our initial step of the data preprocessing method, data cleaning, we continued the process to ensure a data set of high standard. The next process involved data integration. We handled the issue of redundancy in this process. Since all our data is from a single database, we did not encounter the problem of running into data duplicates. Our data reduction was a significant step in our data preprocessing. This process eliminated irrelevant features and reduced noise which helped us crucially since we are working with a massive dataset. In turn, this will speed up the mining and allow for more straightforward visualizations. Our goal for data reduction was to use a dataset much smaller in volume representative of our whole dataset which would produce almost the same mining results as it would for our entire data set (see section 'Create Test Data' below). Our final step of data transformation included using the Pandas library in python to perform operations such as data discretization and normalization.

5.1.7 Create Test Data

We extracted a smaller subset of data from our data set to create our test data set. The test data allowed us to practically run code on a sample before applying it to the original dataset with over 397 million rows of data. Even though we are running an EC2 Instance through AWS that can process through all of our data quickly and seamlessly, we still think it would be beneficial to work through a smaller dataset on our machines as it would still take a substantial time running through our EC2 Instance. Moreover, since the AWS Instance we are running on is significantly more costly than running through the dataset on our computer, it would save us not only valuable time but money as well.

5.1.8 Data Visualizations

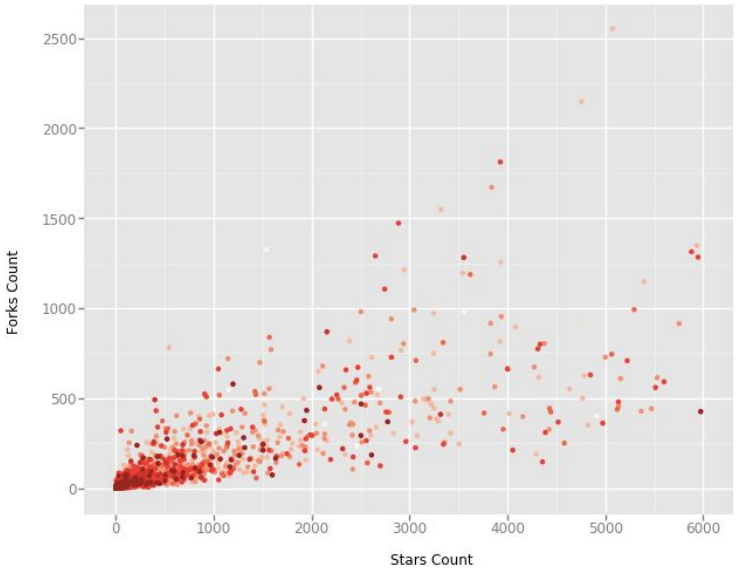
By the 17th of April, we plan on producing a set of stimulating visuals that we have obtained from mining through our dataset. This milestone will allow us to not only understand our dataset but also enable us to display our findings succinctly. We plan on visualizing what our team has stated in our Problem Statement (1) including finding trends within programming languages, popularities of open source repositories over time as well as their life cycles. We also plan on visualizing additional topics we might find interesting while mining the dataset. This additional knowledge will allow us to mine the data adequately given our other domain experience.

5.1.9 Data Analysis

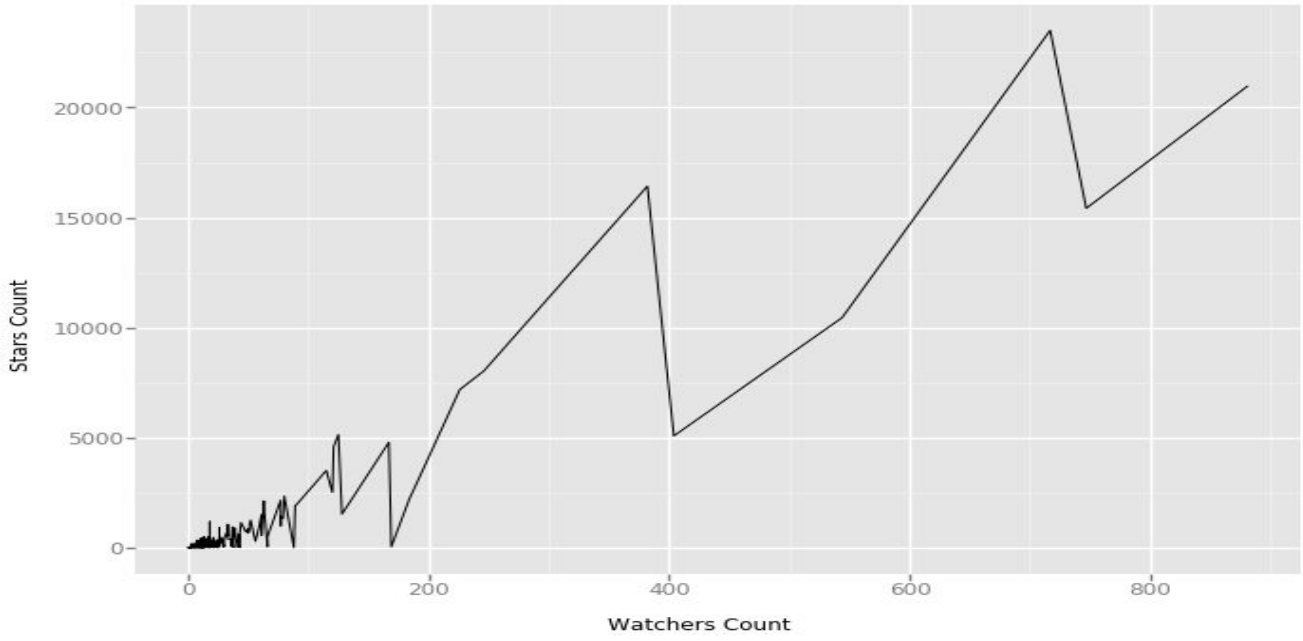
In addition to our data visualization milestone, our data analysis milestone is in place so that we can start to mine and pull real results from our dataset. While all the previous milestones were in place to set us up to do our data analysis, we can finally begin to extract valuable knowledge from our dataset. Our team plans on utilizing many different clustering methods on the

data, and across many different dimensions to identify and find undiscovered patterns about open source development.

We plan on applying most of the skills and techniques we are learning in this class. A rough exhaustive list includes clustering through k-means, k-medoids, DBSCAN; confusion matrices; contingency tables; naïve Bayesian classification; and linear, multiple, and log-linear regression models.



analyzing the data we get to see what kinds of applications or implications that insight can provide. We will attempt to understand our findings through a few different metrics. First, we will evaluate our visualizations and find anything interesting from them. Next, from our data analysis using the tools stated from our Tools section (6), we will see if we can find anything that seems to provide any sort of information our team might find intriguing or unique. Moreover, once we have dug through our



5.1.10 Analyzing Results

Similar to the previous milestone, analyzing our data is in place to allow us to examine our current results and attempt to comprehend all of our findings. While the previous milestone is our team mining the data, in findings, we can further proceed to understand the direction our team would love to move forward with on this project.

5.1.11 Application

Once we complete analyzing our data, we will conclude the applications from our results. This process will help us answer our questions that we defined in Evaluation Methods (5). We initially stated that our data mining would focus on four main areas: tracking trends in programming languages, analyzing how popular repositories have changed over time, how contributions to those repositories have changed over time, and repository life cycles. We will conclude applications and apply the results we will gain through our data analysis to these topics. In our application, we will focus on how our conclusions will help us make better and more useful decisions regarding open source projects. Open source projects play a significant role in the development of software. Tens of thousands of open source projects run worldwide, and millions of users rely on open source software [2]. Concluding applications based on our data analysis will help us better understand open source software that influences millions across the world.

5.1.12 Interactive Site With Visuals

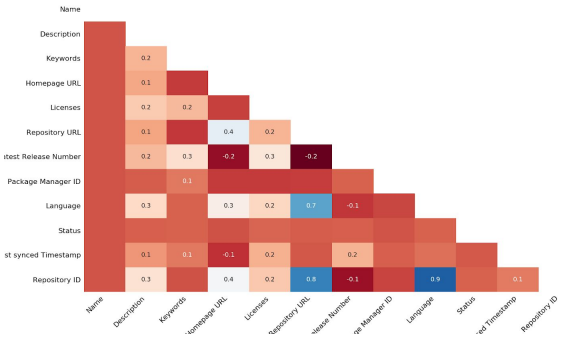
While the course does not require this milestone, we think it would be a great way to showcase the data we have mined. We plan on hosting a website through GitHub pages, as a way to view and interact with the findings our team has mined. We aim to provide all our visuals, analyses we encountered, and how we mined and found all of our findings on our site so others can try and explore and understand how we came to this point. Creating this website is an essential step for us as we want to show off all the hard work we put into this project, as well as allow others to use the knowledge and data we mined to hopefully better the Open Source Software Community.

5.1.13 Final Paper

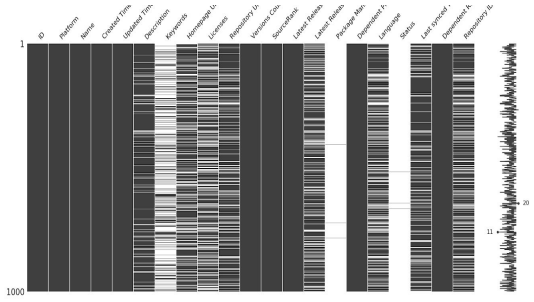
While our course requires this final paper, we still think it is a great way to portray the culmination of work our team has put into the project over the past few months. The final paper will be in the ACM SIG paper format with 11 point font and 1.1 line spacing just as all of our other progress reports are and will include our abstract, all of our related works and findings, as well as the tools we used. Moreover, if our findings are significant, we plan on releasing our paper for others to learn from and extend on if desired.

5.1.14 Results from Preprocessing

Below are a few graphs that parse through the entire dataset and provide us with a cluster map and a frequency table that hopefully visualizes the data in a meaningful way. Most of the other visualizations we have pulled so far have either provided no visually appealing data or did not correlate attributes.



The above graphic is a representation of the "completeness" of our data. The graphic shows the Nullity correlation of one particular data set. The nullity correlation ranges from -1 (if one variable appears the other definitely does not) to 0 (variables appearing or not appearing have no effect on one another) to 1 (if one variable appears the other definitely also does). Essentially this gave us a quick overview of the "completeness" of the data and allowed us to gain more insight into our dataset.



The graphic presented above shows us the "completeness" of one of the data sets. On the top of the graph you can see each column in the dataset. A dark bar represents complete data, where whitespace indicates null values. This quick visualization of the data allowed us to easily rule out points of data to try to use in our data mining process as there seems to be a lack of information on some columns.

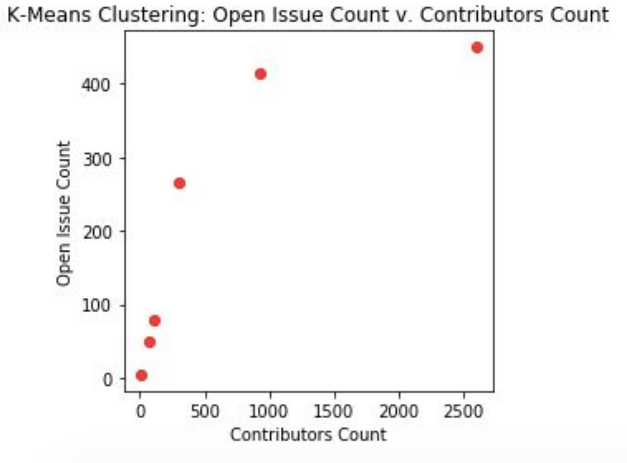
5. 2 Data Warehouse

After getting the data from libraries.io in a much more usable form, we then loaded the dataset into our EC2 instance. The server acted as our data warehouse where we were able to interact and hold all the data in a meaningful way. Through AWS, we were able to manipulate the data in any way we wanted. And since the data we have was static, we didn't have to worry about data coming from a transactional database.

As for the data marts, we used individual aspects of our analytical work. For example, we took data from the primary dataset and created smaller subsets. These smaller subsets were necessary as it allowed our team to analyze the data more efficiently. And since the larger dataset was much more computationally difficult to compute, this was one of the only options we seemed to find.

5. 3 Clustering and Classification

The use of clustering was essential to answering a few of our core questions. To try to find deficiencies in the open source community we thought it would be best to try to cluster repositories, and from there we could identify cluster that are "lacking" and can befinift from extra community involvement. To do this we went about implementing a K-Means clustering algorithm on the dataset. We can use common metrics from each repository to attempt to cluster repositories based on their "health". We define a "healthy" repository to be a repository that has a comparable level of contribution to the issues that the repository is facing. Running K-Means on the Repositories data set, using "Open Issue Count" and "Contributors Count" as the X and Y axis, and using six clusters, returned interesting results.



Using the center of each cluster we can compute the average number of Open Issues/Contributors. This can give us an insight into the "health" of each cluster.

Cluster Number	Open Issues/ Contributors
1	1.28
2	3.71
3	3.49
4	5.78
5	1.03
6	0.26

Given the results defined above, we can see that cluster 4 is in most need of more help, as the have the highest average ratio of Issues to Contributors. Not only did we run K-Means on just these two dimensions, we ran it across multiple metrics. We analyzed Six different metrics for each repository: "Stars Count", "Forks Count", "Open Issues Count", Watchers Count", "Contributors Count", and "SourceRank"(Libraries.io custom ranking for each repository). The full results of all these clusterings,

including the center of each cluster, and the number of repositories in each cluster, can be found here.

KMeans was not the only clustering algorithm we ran on the dataset. We implemented DBSCAN as well. When we ran DBSCAN on a smaller subset of our data we found that the results were inconclusive. The clusters that the algorithm returned were indistinct and had little variation between the two. We believe that this is caused by the massive amount of data we have, as well as the fact that most repositories have very little distinction between them. When we ran DBSCAN on a larger set of our data we had lots of issues with RAM and memory management on our EC2 and we were unable to extract results. Although we were disappointed with this outcome, we learned a lot about the differences between clustering algorithms and how each one can produce massively different results.

6. Key Results

To revisit our key questions: What insights can we gain to improve the open source community further? How can we identify areas in the open source community that need improvement? Can we predict upcoming popular repositories?

How can we identify areas in the open source community that need improvement?

The answer to this question, we believe that just one cluster on two data points is not enough. We decided to compile a list of repositories that we believe need help, but also are worth working on. We decided on the metrics of "Issues/Contributors", "Stars/Contributors", and "SourceRank/Forks". Using our K-Means algorithm we can detect the repositories most in need of contribution help, those with the most interest from the community, and Libraries.io's favorite repositories with the least amount of work.

Cluster Number (Repository Count)	Open Issues / Contributors
1 (6390)	1.28
2 (703)	**3.71**
3 (112)	3.49
4 (20)	5.78
5 (143696)	1.03
6 (258)	0.26

We selected Cluster 2 as the most "unhealthy" repository from this group. We acknowledge that Cluster 4 is in more "need" of community help, although the small sample size of repositories make the cluster more of an outlier.

Cluster Number (Repository Count)	Star Count / Contributors
1 (306577)	7.06
2 (277)	**81.62**
3 (13)	84.89
4 (1024)	70.04
5 (62)	83.13
6 (4554)	43.69

Here we will select Cluster 2 as we view it as the largest cluster in need of community assistance.

Cluster Number (Repository Count)	Source Ranks / Fork Count
1 (230669)	2.90
2 (511)	151.44
3 (31)	806.03
4 (1)	2423.04
5 (111)	**370.19**

6 (3453)	36.04

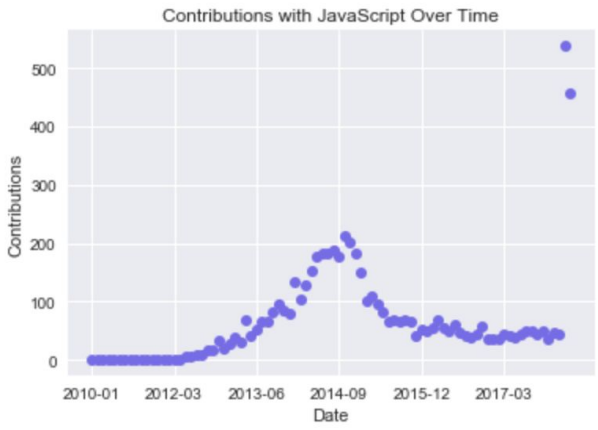
Note: Cluster 4 seems to be a major outlier, KMeans should have been reran with just 5 clusters and not 6.

Here we will select Cluster 5 as we view it as the largest cluster with the highest rank and least amount of contribution.

Merging the list of these three clusters we were able to identify 17 repositories in all three clusters. We deem these 17 repositories to be in need of the most community assistance. Using a more detailed approach we believe that our data mining could produce a list of top repositories that could use contribution. From there we could publish that list and hopefully attract developers to contribute to those repositories.

Can we predict upcoming popular repositories?

We took the same approach as the previous question to answer this question. We identified the clusters of "Stars/Contributors", "Stars/Forks", and



"Forks/Issues" as the best indicators of future growth and popularity of a repository. This will give us a list of "up and coming" repositories that we believe will become very widely used and popular soon. We do note that this approach may be flawed. The best way of predicting future "success" and prominence would most likely use Machine Learning to analyze past trends and predict future ones. This approach however is far outside of the scope of this class, so we deemed our method, the "most accurate" for the scope of this class.

Cluster Number (Repository Count)	Star Count / Contributors
1 (306577)	**7.06**
2 (277)	81.62
3 (13)	84.89
4 (1024)	70.04
5 (62)	83.13
6 (4554)	43.69

Cluster 1 provides us with a very large sample of repositories with a low "Star/ Contributors" ratio, meaning there is lots of development support behind the repositories given their respective interest.

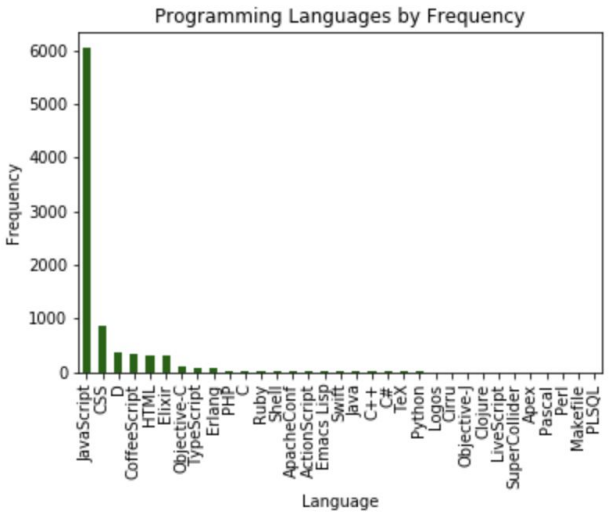
Cluster Number (Repository Count)	Star Count / Fork Count
1 (206336)	**3.45**
2 (243)	5.08
3 (13)	3.70
4 (4258)	5.18
5 (60)	4.23
6 (939)	5.65

Cluster 1 provides us with a very large sample of repositories with a low "Star/ Fork" ratio, meaning there is lots of development being worked on behind the repositories given their respective interest.

Cluster Number (Repository Count)	Fork Count / Open Issues
1 (122622)	2.92
2 (30)	22.16

3 (108)	11.39
4 (1)	131.86
5 (482)	**7.95**
6 (3147)	5.55

Note: Cluster 4 seems to be a major outlier, KMeans should have been reran with just 5 clusters and not 6.



Cluster 5 provides us with a decent sized sample of repositories with a large "Fork/Issue" ratio, meaning there is not a lot of errant code that is committed to the repositories.

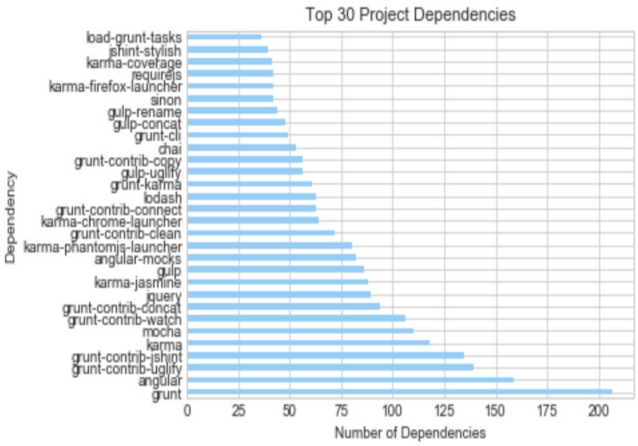
The subset of repositories that span these three clusters number 348. These repositories, we believe, have the potential to be very large and heavily contributed to repositories in the future. As noted above, we know this is not the best way to predict trends and future growth, but limited to the scope of this course, we are proud of the "predictions" we made.

Overall we were able to use clustering to identify sets of repositories to answer our initial questions.

7. Applications

7.1 Benefits for the Software Community

By analyzing time-series of different programming languages, we can display numerous attributes that can not only be applied open source software developers but general coders. Identifying that repositories using Python or Java are currently experiencing very high contributions, and showing how trends between dying and popular programming languages can help prove the likelihood of these



languages dying out soon or continuing to prosper.

7.2 Future Repository Popularity Detection

Given our research and analysis Through our cluster analysis, we were able to identify that "Stars/Contributors," "Stars/Forks," and "Forks/Issues" were the best indicators of future growth and popularity of a repository. These relationships will help developers working in open source software a general direction to work towards when creating new software. Furthermore, understanding the "Fork/Issue" ratio shows tremendous potential in future heavy traffic and contributions.

8. Visualization

While we did an extensive job to portray a sufficient amount of data visualizations within this paper, we curated a website that enables users an interactive way to play around with the findings stated throughout this document. Our site has many other different aspects including access to this paper and a separate page showcasing our team be found here.

9. References

[1] “Stack Overflow Developer Survey 2017.” Stack Overflow, insights.stackoverflow.com/survey/2017.

[2] Scacchi, Walt. “The Future of Research in Free/Open Source Software Development .” 2010.

[3] Hu, Yan, et al. “Influence Analysis of Github Repositories.” SpringerPlus, Springer International Publishing, 5 Aug. 2016, www.ncbi.nlm.nih.gov/pmc/articles/PMC4975729/.

[4] Chełkowski, Tadeusz, et al. “Inequalities in Open Source Software Development: Analysis of Contributor's Commits in Apache Software Foundation Projects.” PLOS ONE, Public Library of Science, 20 Apr. 2016, journals.plos.org/plosone/article?id=10.1371%2Fjournal.pone.0152976.

[5] libraries.io/data.

[6] David Hand, Heikki Mannila and Padhraic Smyth. 2001. “Principles of Data Mining”, (34-37). MIT Press.