GitPrime

The Fundamentals

October 26, 2017

Contents

1 Introduction **1**
1.1 Context, motivation, and objectives 1

2 Author segments **2**
2.1 Data composition 3
2.2 Total active days 3
2.3 Author-day commits distribution 3
2.4 Notes on commits per day 7
2.5 Understanding the significance of commits 9
2.6 Author segmentation 10
2.7 Author behavioral pattern analysis 12
2.8 Author label calculation 13
2.9 Meaning making of labels 15

3 Commit benchmarks **15**
3.1 Commits per day 15
3.1.1 Summary 16
3.2 Commits per week 17
3.2.1 Summary 17

4 Active days benchmarks **17**
4.1 Active days per week 18
4.1.1 Summary 19

5 Impact benchmarks **19**
5.1 Impact per week 20
5.1.1 Summary 20

6 Efficiency benchmarks **21**
6.1 Efficiency per week 22
6.1.1 Summary 22

7 Conclusion **24**

1 Introduction

This document is intended to provide supporting statistical evidence for GitPrime's Fundamentals and its associated Leadership Playbook. The primary audience of this document is an active statistician practitioner with a deep working knowledge of both traditional computational statistical analysis and familiarity with the more current industry trends and thinking as it relates to looking at, communicating, and interpreting statistical evidence.

1.1 Context, motivation, and objectives

Statistical evidence and probabilistic reasoning today play an important and expanding role in software development, not least in relation to the processes and methodology for commercial software development. It is vital that executives, managers, team leads, and developers involved in managing large commercial teams of software developers are able to comprehend and deal with probability and statistics appropriately. There is a long history and ample anecdotal evidence with misunderstandings relating to statistical information and probabilities which have lead to fallacies, flawed deductive reasoning, and management misdirection that at least prove to be a nuisance to practicing commercial software developers and at worst sink promising software projects.

Software management practice, as we idealize it, is strongly wedded to the principle of fact finding by management and practitioners employing their ordinary common sense reasoning. Notwithstanding the unquestionable merits of technically skilled management involvement in commercial software development, it cannot be assumed that leadership are equipped by their general education to cope with the analytical demands of statistics or probabilistic reasoning. This predictable deficit underscores the responsibilities of those using statistical tools for productivity gains, within the broader framework of modern software workflows, to present statistical evidence and probabilities to both management and practitioners clearly. Yet professional managers' grasp of statistics and probability may be little better than the average full-stack developer. More surprising, even researchers and analysts, whose expertise is typically the immediate source of statistics and probabilities presented in proposed frameworks and processes, may also lack familiarity with relevant terminology, concepts, and methods. Leadership must satisfy the threshold test of competency before being allowed to unilaterally drive soft-

ware development using homegrown tools and career-scoped data.

GitPrime, and specifically the Fundamentals and the Data-Driven Playbook, is designed to provide a comprehensive toolset grounded in statistics and wedded in common sense reasoning. We provide every effort to make our tools accessible and usable day-to-day.

In preparing this document, the researchers at GitPrime studied over 7 million anonymized redacted commits across nearly 88,000 authors.

(1) Overview - the major thread of this paper is an exploration of the difference in benchmarking scores across author segments.

- We argue that author segments exist. (section 2).
- We use behavioral patterns and build a segment and a similarity score to label authors. (section 2.6).
- We benchmark four metrics for our authors:
  1. Commits (section 3)
  2. Active Days (section 4)
  3. Impact (section 5)
  4. Efficiency (section 6)

(2) Time period notes - we select an initial time periods over which to view these variables.

- Most of these initial choices were made to get a preliminary sense for categorization of information.
- We may eventually see cyclic patterns that push us to consider other time periods and build models that account for such things as seasonality.

2 Author segments

Starting with a data set that ranges across all of 2016, we explore each author's count of commits per day. Since each author can have anywhere from

0 to 365 days of committing in a year, we say there is up to 365 *active days* (i.e. a day an author made one or more commits) available for each author.

2.1 Data composition

We see the following counts for GitPrime's 2016 dataset.

- total commits: 7,042,280
- total author-days: 1,855,352
- total authors: 87,706

2.2 Total active days

An active day contains one or more commits per author. As can be seen in Figure 1, most of our authors have less than 100 active days. In fact, looking at Table 1 on page 4, we see that 90 percent of our authors make commits on fewer than 90 days in an entire year.

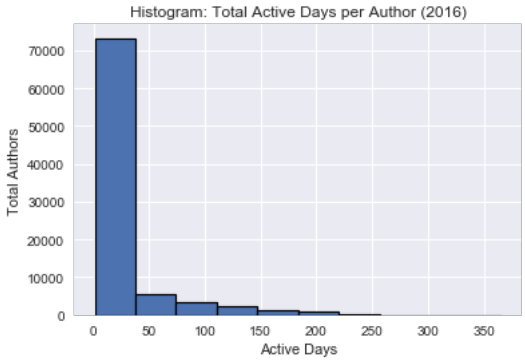

Figure 1: This histogram shows the total days that each author had one or more commits.

2.3 Author-day commits distribution

We only include non-zero daily commit counts for these metrics to specifically explore the behavior of authors on days when they are committing code. For

Percentile	Active days
10th	1.0
20th	1.0
30th	1.0
40th	2.0
50th	2.0 (MEDIAN)
60th	4.0
70th	10.0
80th	26.0
90th	73.0
91th	81.0
92th	90.0
93th	99.0
94th	110.0
95th	122.0
96th	136.0
97th	154.0
98th	175.0
99th	203.0
100th	365.0

Table 1: The percentiles for active days per author

Percentile (author-days)	Commits per day
10th	1.0
20th	1.0
30th	1.0
40th	2.0
50th	2.0 (MEDIAN)
60th	3.0
70th	4.0
80th	5.0
90th	8.0
91th	9.0
92th	9.0
93th	10.0
94th	10.0
95th	11.0
96th	12.0
97th	14.0
98th	16.0
99th	21.0
100th	975.0

Table 2: The percentiles for commits per day for all author days.

the corresponding commits per author-days, we see the following percentiles. Note the long tail evident in the percentile distribution in Table 2 on page 5. The middle 50 percent of the data lives in the interval [1,4] commits per day as shown in the box plot of Figure 3 on page 6. Regarding the middle of this commit data, we calculate bootstrapped confidence intervals:

```
95% confidence interval for median author-days: (1.99, 2.01)
95% confidence interval for mean author-days: (3.78, 3.80)
```

In the following discussion, we move away from using these intervals and in doing so argue that we improve our understanding of author behavior.

Figure 2: The box plots shows commits per day for all author days.

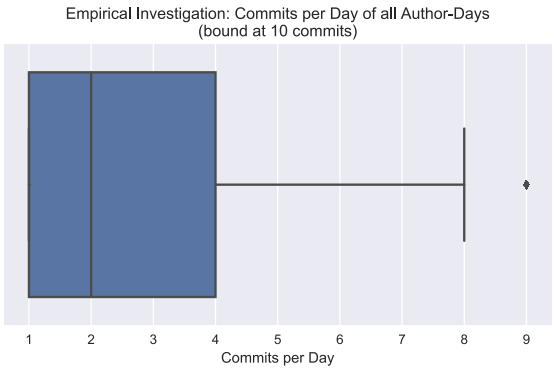

Figure 3: The box plot shows commits per day for all author days bound at 10 commits per day.

	day1	day2	day3	day4
Sue	0	2	0	3
Tom	7	73	2	1

Table 3: This table is an example of how the author-days commit distributions contain unequal counts of commits per author.

2.4 Notes on commits per day

- Each day contains commit contributions from numerous authors, but those authors' multiple scores are compared to both other authors and themselves. This issue is resolved later in the paper, but to understand the context, we will next consider an example.

- For example, examine two hypothetical code contributors, Tom and Sue (see the data in Table 3 on page 7). Sue has two author-days, [2,3] and Tom has four [7, 73, 2, 1]. The union of Tom and Sue's data then is [2, 3, 7, 73, 2, 1], which is the set of data from which their author-day percentiles would be built (i.e. all non-zero values).

- Notice that Sue has two values and Tom has four values included in the set: [2, 3, 7, 73, 2, 1]. It's important to note that when we collect data this way, we are no longer comparing author to author. Instead, we're comparing author-days to author-days—a small but vital difference for understanding the methodology that we develop below. Since authors can have an unequal number of values contained in the set, the above percentiles show the number of commits in author-day aggregations—does this matter for the meaning of the results? The meaning is suspect and possibly hard to digest because the total number of values per author varies wildly; hence, if we did not change course, then author behavior is not the focus of this study. Instead, we're investigating the author-days. We'll clean up this conceptual problem later, but for now, the goal is to just clarify the issue.

- As noted in the commits per author-day percentiles (Table 2) and histogram (Figure 4), the majority of author-days contain fewer than 25 commits per day. But there's a long tail of commit counts per author-day that extends to roughly 1,000 commits per day.

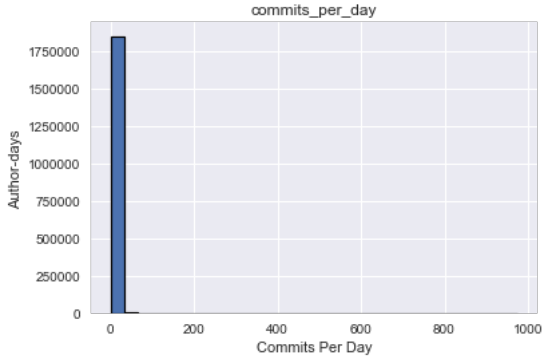

Figure 4: This not-so-descriptive histogram extends up to 1,000 commits per day because it contains a long tail.

- What is the average value in this histogram (see Figure 4 on page 8)?
  - To find the middle of the Figure 4, we can see from the percentiles in Table 2 on page 5 that the median commits per day in author-days is 2.0.
  - Is the above median of 2.0 actually meaningful? No. We can not place any sort of confidence in this statistic because it doesn't come from a distribution that we know how to measure. Then why the confidence interval for mean and median measurements above? The above confidence intervals exist because we derive them from bootstrap samples and examine the sample's mean of means to declare, "From this data set, we're certain that the median is within this range." Unfortunately, this statement does not imply that the median has real meaning. It simply tells us that there is a tight band circling the median such that the calculation is consistent across many samples.
  - What could we do instead of using a single measure of central tendency to describe the center of this data? One idea is to describe where a large chunk of the data lives. Recall that the middle 50 percent of the data resides in the interval [1,4] commits per day (see the box plot in Figure 2). However, we'll work through a more detailed process to improve this description.
- How varied are these author-days?

- The standard deviation of commits per author-day is over twice the size of the median: 5.61 commits per author-day, which is impressively large given that our data set median is 2.

2.5 Understanding the significance of commits

In the reading above, we outlined a problem. We now begin to unravel our new approach to solving this problem. In the end, we advocate that we have created a more robust description of our data.

- What is the importance of author-days?
  - Understanding author-days is helpful because it provides a sense of frequency. This is the core of the new approach. The author-days inform this approach via frequencies (i.e. how often the total number of each authors' commits per day fall into a specific range). We transform author-days into one frequency score per author for each commit bucket. Using proper segmentation techniques (see pandas qcut), the following buckets emerge for commits per author-days:
    - [1.0, 2.0] commits
    - [3.0, 5.0] commits
    - [6.0, 1000.0] commits
  - To distill author-day information from many values into a single value for each author, the focus of the new method that we develop is to consider their daily commit frequency in each of the above buckets. These frequencies provide us with a quantified behavioral pattern that is then comparable across authors. We then cluster these patterns into three segments below.
- Why do these segments matter?
  - Rather than say that certain data from individuals must be excluded from the analysis, we can state that our author segments have a certain commit cadence and let the reader do what they want with that information. Hence, we're including all of the data and letting the reader determine the significance rather than making some sort of cut point for .

- Similar to any classification attempt between active or inactive authors based on an omission method from an author's rolling average of commit counts, we calculate and calibrate commit metrics for each segment. These segment metrics are then translated into something usable for GitPrime's products.
- What do these segment metrics look like? Read below to learn about more of the analysis.

2.6 Author segmentation

Our objective here is to answer this question, "How often does each author commit between [1 to 2], [3 to 5], and [6 to 1000] in a day?" The frequencies (or percentage of time in each commit bucket) provide us with our user behaviors.

Without an additional filter, however, we'd again just be looking at one metric such as a mean or median of the entire set—an ideal solution if the data is consistent across all authors. But in our case, we have different behaviors for different author segments, which is why our histograms are so far from looking like normal curves. Through many trials with different variables, it's apparent that we can isolate author behavior with author segmentation. We are using three author segmentations: Occasional Contributors, Primary Contributors, and Leading Contributors. These new author labels are developed from two considerations: commit frequency across time buckets and active days per year. Aggregating average behavior over active days per year, we see distinct patterns emerge for each segment:

- Occasional Contributors in Figure 5
- Primary Contributors in Figure 6
- Leading Contributors in Figure 7

We can summarize our results at this point and consider setting the stage to investigate metrics. Summary points are listed below.

- Focusing on a single central tendency measure for the distribution of commits per day glosses over the complexity of the information. The variation in this dataset is large enough that using a simple measure of central tendency detaches meaning and value from the statistic.

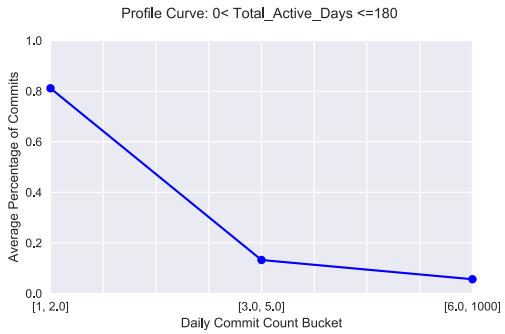

Figure 5: This profile curve shows the details for the frequency of the Occasional Contributors.

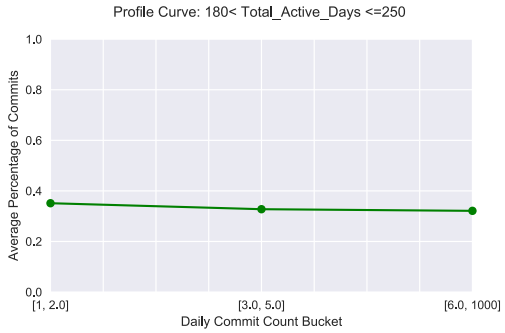

Figure 6: This profile curve shows the details for the frequency of the Primary Contributors.

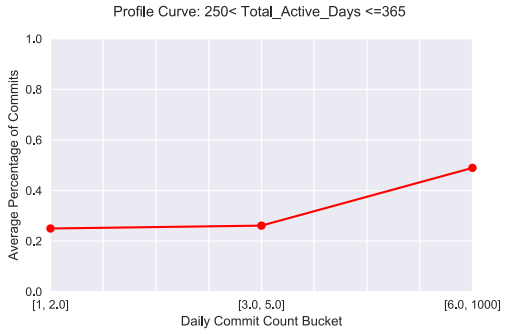

Figure 7: This author segmentation curve shows the details for the frequency of the Leading Contributors.

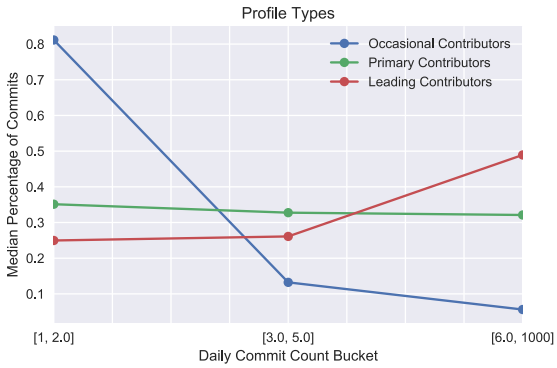

Figure 8: These are the curves that describe the typical behavior of someone in each category.

- The actual calculation for the variation is notable: in most cases, the standard deviation is about the same or even twice the size of the median.
- What can we do with these author segmentations?
  - We can score any author as a distance from the segments and then take the closest value as the label for the group. In summary, we can now identify authors by their behavioral type.
  - We label each other type as follows:
    1. Occasional: lowest level of contributors
    2. Primary: mid-range contributors
    3. Leading: highest level of contributors
  - With these added labels, we explore central tendency per segment.

2.7 Author behavioral pattern analysis

To better understand our users, we examine each author behavioral pattern and assign a nearness score to one of three author segmentations described above. Our three author segmentations describe the typical author behavior patterns in Figure 8 on page 12.

Here, we consider three real-life author behavior patterns to show how we label authors into segmentation groups. We can glance at the actual frequency data of each authors committing behavior (see Figure 9). Plotting their behavior produces the curves in Figure 9. To summarize the process, we take non-labeled authors, examine their behavior, and then color their actions based on its nearness to an author segmentation (explanation of that calculation is below in section 2.8). This gives us greater insight into our users. These author behavior curves still tell us commit frequency, but we have an added label now that allows us to cluster the authors. Quite simply, we've built a way to score individual authors similarity to the author segmentations (see Figure 10).

real_author_id	70601	70794	70820
org_id			
[1, 2.0]	1.0	0.428571	0.260116
[3.0, 5.0]	0.0	0.428571	0.323699
[6.0, 1000]	0.0	0.142857	0.416185

Figure 9: These are real examples of author behavioral patterns, not averages. Each column is an author and the percentage of time that they spend in each bucket is listed in the rows. This behavior pattern is eventually compared to the author segments and the closest match becomes that author's label.

2.8 Author label calculation

In this ongoing discussion, we continue to separate the segments with color:

- red: Occasional Contributor
- green: Primary Contributor

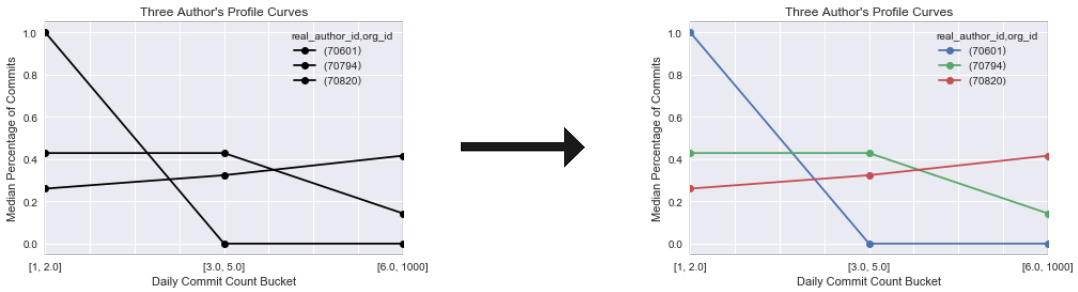

Figure 10: The scoring process measures each author's behavioral pattern and produces a label based on a similarity score to the typical author segmentation of each category (see Figure 8 for the segmentation analysis).

- blue: Leading Contributor

We label authors as Occasional, Primary or Leading by prescribing a similarity score to author behaviors across each segment. Much like how kmeans finds points near a centroid, the author's segment is determined by the closest similarity. The majority of our authors are Occasional Contributors, which might be expected from the initial findings discussed above in section 2.2 that specifically addresses the results in Figure 1. The total number of labeled authors per segment are listed in Table 4.

Segmentation	Total Authors
Occasional Contributor	68,848
Primary Contributor	15,256
Leading Contributor	3,602

Table 4: Total counts of authors for each segmentation.

To consider the methodology in practice, we can see in the table below that each author receives a similarity score to each of the three segments.

real_author_id	org_id	Occasional Contributors	Primary Contributors	Leading Contributors	profile_type
70565	[redacted]	0.932197	0.477629	0.333354	Occasional Contributors
70566		0.945564	0.428146	0.285241	Occasional Contributors
70601		0.762900	0.205470	0.066958	Occasional Contributors
70701		0.911271	0.433651	0.261065	Occasional Contributors
70794		0.508192	0.781028	0.575601	Primary Contributors
70819		0.356780	0.918886	0.857179	Primary Contributors
70820		0.314302	0.868232	0.903119	Leading Contributors
70821		0.820796	0.407325	0.296205	Occasional Contributors
70822		0.208076	0.698422	0.905241	Leading Contributors

2.9 Meaning making of labels

To add value to our analysis, labeled authors must show distinct patterns when aggregated into the segmentations. If the segments show identical metrics, we have not yet distinguished author behavioral patterns. Take note that we move beyond a method description here and now seek meaning in our new method. Instead of measuring the mean or median value as a single measure of central tendency across author-days, we highlight that there are segments of detectable behavior within our data. We are thus focused on segment-specific metrics below.

3 Commit benchmarks

We consider both commits per day and commits per week across the three author segments in the analysis below.

3.1 Commits per day

If we consider all author-days, we see in Figure 11 on page 16 that the distribution is skewed right, pulling the mean values past the median scores. If we treated all authors as one group, the analysis would conclude that the average commits per day is somewhere between two and four commits per day. However, our use of author segmentation tells a more detailed story.

Using our new method, in Figure 12 on page 16, we discovered that the difference between author segmentations is vast: note the distinct bands

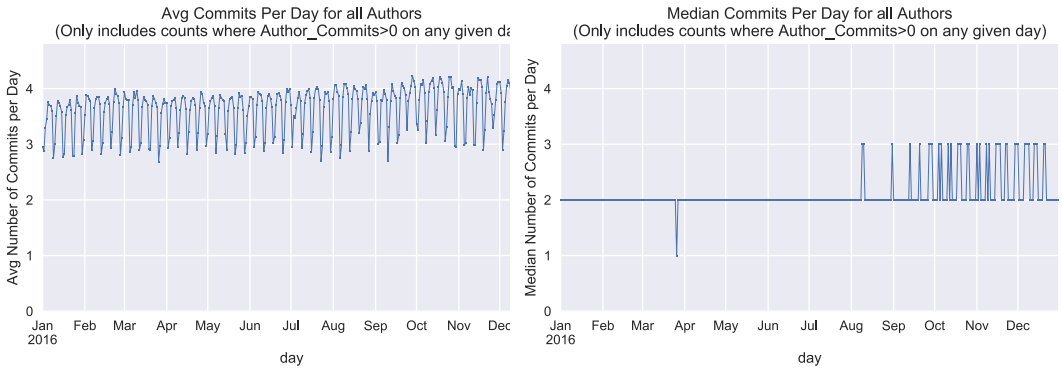

Figure 11: If we merely took the mean/median of the commits per day across all author-days, we would get these unspecific daily values.

that represent Occasional Contributors, Primary Contributors and Leading Contributors in 2016.

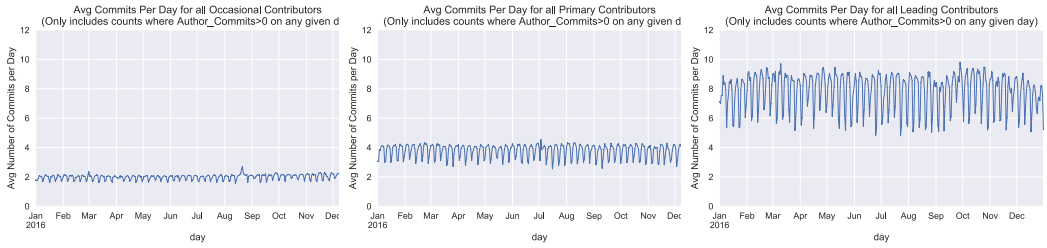

Figure 12: Using the new method, the average commits per day across these three groups is quite distinct, which means that we see a notable behavioral difference between the groups.

3.1.1 Summary

Three groups of individuals based on commit activity have, on average, daily counts around 2, 4, and 8. But author segmentation was created because user behavior wildly varies. Our data examination describes how many commits people make on days when they make at least one commit:

- Occasional contributors - around 2 commits per day.
- Primary contributors - around 4 commits per day.

- Leading contributors - around 8 commits per day.

The question may arise as to about what we can say with confidence about any of these numbers (great confidence interval discussion here). We aim to bound individual behavior—which could inherently be wrong for any individual given that nothing stops someone from being a Leading Contributor one day and an Occasional Contributor the next. Our labels serve as an at-a-glance author pattern. If we can avoid saying that someone with Leading Contributor level of commits per day is superior to those with Occasional Contributor levels, that might leave room to inject factors beyond volume. For instance, does a Leading Contributor have higher impact? We dive deeper into group attributes or lack thereof in the following sections.

3.2 Commits per week

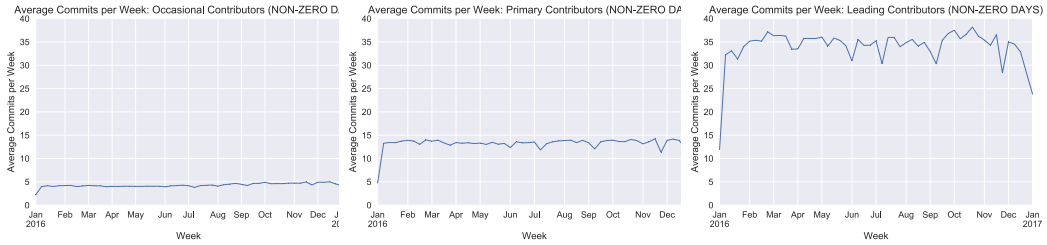

Figure 13: Average commits per week.

3.2.1 Summary

In summary, we expect to see Occasional Contributors hover around 5 commits per week, Primary Contributors to have slightly over 14 commits per week, and Leading Contributors to reach 35 commits per week. See Figure 13 for details.

4 Active days benchmarks

Recall that we define an active day as a day in which an author has one or more commits.

4.1 Active days per week

In Table 5 on page 18 we consider the active days per week percentiles for all authors-days and note the median of 2 active days. We also note the shape of the distribution in Figure 14 on page 18 with a right skew. We'll see more details of this distribution once we dive into the analysis below, but we can certainly see distinct variation across each author segmentation in Figure 15.

Percentile	Active days per week
10th	1.0
20th	1.0
30th	1.0
40th	2.0
50th	2.0 (MEDIAN)
60th	3.0
70th	4.0
80th	4.0
90th	5.0
100th	7.0

Table 5: Percentiles of active days per week for all author-days

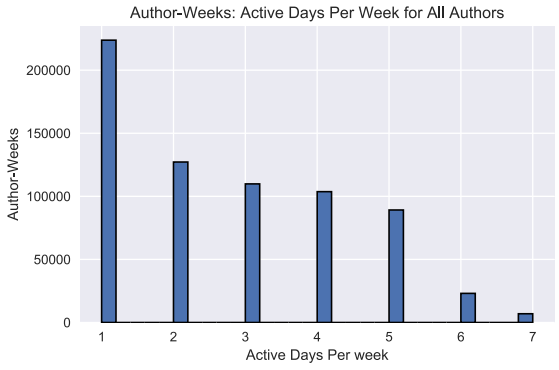

Figure 14: Active days per week for all author-days.

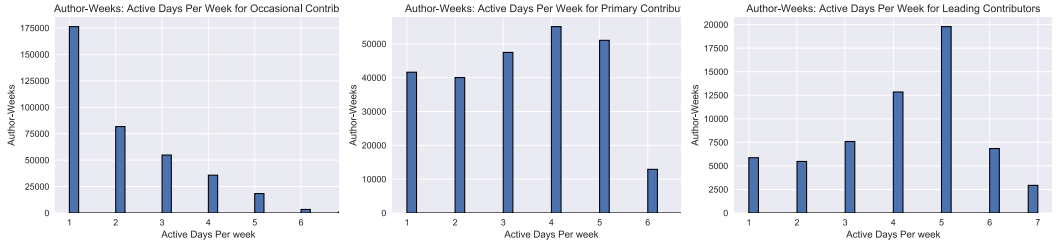

Figure 15: For all author-days in each segment type, we can get a sense of the variation of active days per week in these histograms.

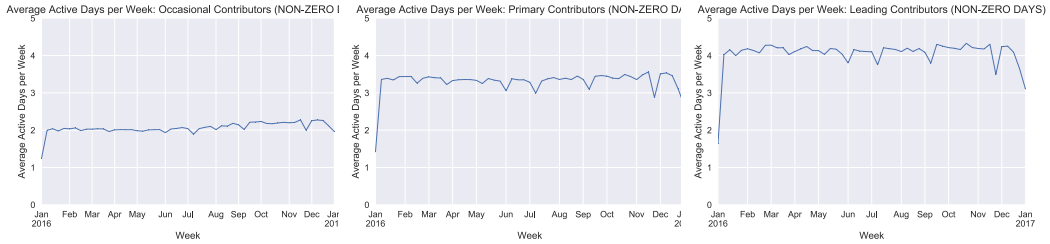

Figure 16: Using the average active days per week for all author-days in each segment type, we get a sense of the stability of the average active days per week over time.

4.1.1 Summary

We expect Occasional Contributors to hover around 2 active days per week, Primary Contributors to be slightly over 3 active days per week, and Leading Contributors to be stretching for 5 active days per week with an average just above 4.

5 Impact benchmarks

Impact is a score composed of two parts. The first part is specialized, but essentially it's a ratio of lines inserted to lines deleted. The second part is a research-based scaling factor that accounts for the deviation from what is considered to be a "normal commit", which is much more than a line count. The product of the two values illuminates the complexity of the commit.

5.1 Impact per week

With a median impact of 81, the middle 50 percent of the author-week impact scores range from 20 to 212—but there is an incredibly long tail that broaches 6,000. Figure 17 shows the long tail in the left-most box plot; the right-most box plot is a truncated version of the other, allowing us to see more clearly where the middle 50 percent of the data lives.

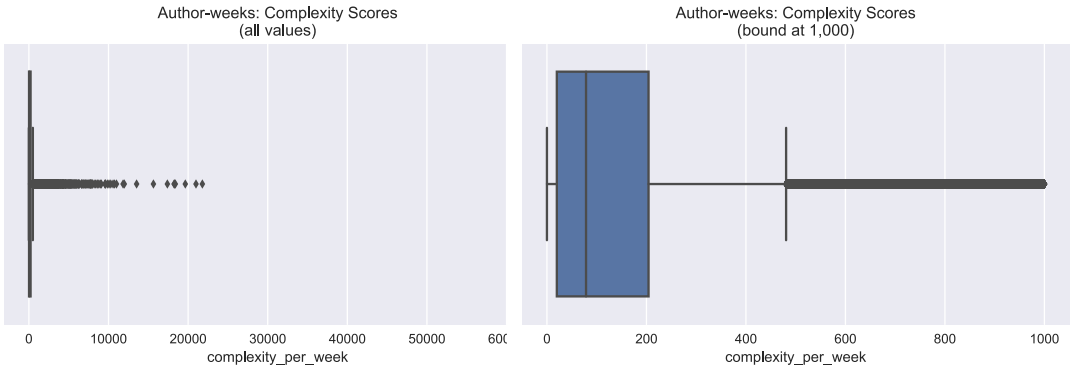

Figure 17: Even though many of the commits have an impact score less than 200, we see a long tail in this distribution.

The weekly variation away from each author's mean value is large, meaning a "typical" week is atypical, that is, far from the author's mean. The middle of Figure 18 identifies that authors generally vary their weekly impact scores by 80 percent of their mean (on average), which is a massive swing week over week.

The average weekly impact score for each author segmentation presents distinction between the groups. See Figure 19, to compare distinct impact score jumps in each group.

5.1.1 Summary

There is large variation in week over week author impact scores. The commit author segmentations have distinct impact profiles, which indicates a likely strong relationship between commit volume and impact score. An impact score over 100 would be significant for Occasional Contributors, whereas

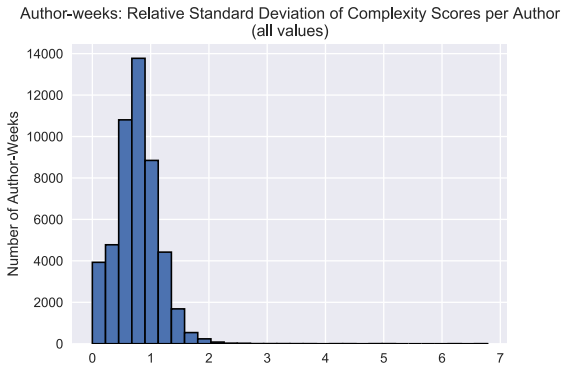

Figure 18: The variation from the mean impact score per author is fairly large.

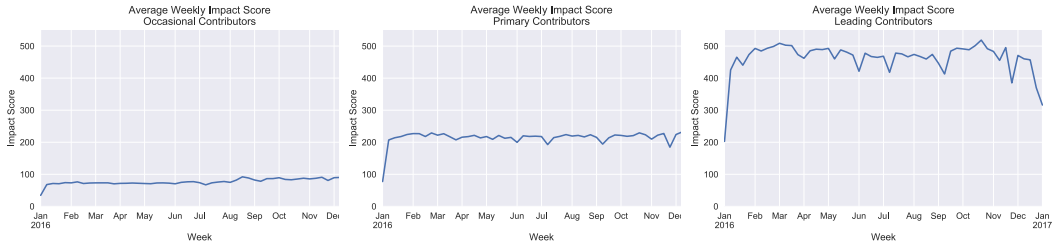

Figure 19: Using the average active days per week for all author-days in each segment type, we see the stability of the average impact scores over time.

Primary Contributors are shooting for scores over 225. Leading Contributors are exceptional with impact scores over 550.

6 Efficiency benchmarks

Efficiency scores try to answer this question: "how much of an author's work is not re-writing their own code?" Answering this question relates directly to the concept of churn—the amount of an author's work that is self re-written. In this analysis, we look at weekly efficiency scores per author segmentation group, but settle on a scale suggesting levels of normality.

6.1 Efficiency per week

According to our research, high levels of churn should not only be easy to detect, but also quite rare. Author-week percentiles (see Table 6 on page 22) show that almost half of all efficiency scores are 100 percent (meaning no churn at all). Further, in the Figure 20 on page 23, there's only slight variation in the efficiency scores across the author profiles—all of which hover near 80 percent efficiency on average. But a quantile-based discretization function can be used to break this data into categories. Using this tool, we find a small difference when comparing Leading Contributors to the other author segmentation groups. It might seem surprising that the Leading Contributors have lower efficiency scores; however this is not all that surprising because this group writes the most code and likely have the largest opportunity to rewrite it. In the efficiency summary, we detail differences between these scales more fully and converge on a single scale. The main point from this section is that because efficiency scores are typically high across all groups, when a work flow starts to include high levels of churn (low efficiency), the deviation from the trend would be noticeable.

Percentile	Efficiency per Week
10th	52.29
20th	70.39
30th	81.48
40th	89.34
50th	95.11
60th	98.81
70th	100.0
80th	100.0
90th	100.0
100th	100.0

Table 6: Percentiles of efficiency per week for all author-days

6.1.1 Summary

Occasional Contributors have the highest average efficiency scores while Leading Contributors have the lowest average efficiency scores. This dichotomy

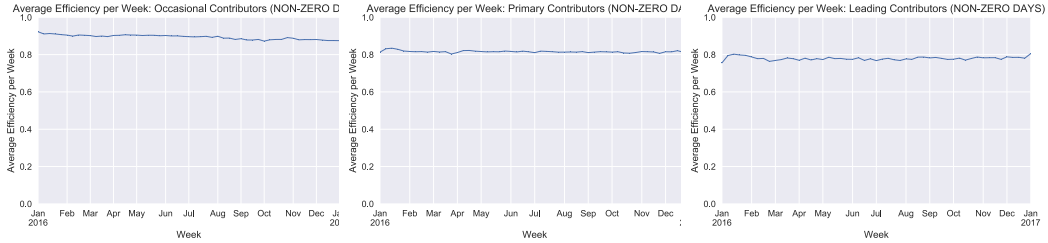

Figure 20: Using the average efficiency scores per week for all author-days in each author segmentation, there is only minor variation between the groups.

is likely due to the fact that Leading Contributors write more code, which increases the opportunity to re-write code.

Three efficiency buckets emerge for all authors scores between 0 and 1:

- (0, 73]: low
- (73, 90]: avg
- (90, 100]: high

For more seasoned coders who would naturally spend more time rewriting their code, we see these efficiency buckets:

- (0, 69]: low
- (69, 86]: avg
- (86, 1.0]: high

We combine these two notions on efficiency buckets to develop a single readable scale seen in Figure 21 on page 24. To build this single scale, we first looked at the endpoints for each section in the above scales and rounded down from the range of the midpoint across the two categories. This single scale suggests that authors with more than 13 percent churn (i.e. less than 87 percent efficiency) may be moving away from higher performance. Further, those authors that start developing a churn rate greater than 30 percent (i.e. less than 70 percent efficiency) are considered to be entering into the lowest-performing category. If we want to draw a hardline with a single number, then we suggest any efficiency score below 70 percent is

symptomatic of other potential issues; whereas, anything above 70 percent could possibly be ignored as typical performance. In summary, this single scale is a tool to consider how we might interpret our efficiency scores for engineering performance. In practice, efficiency should be used in conjunction with impact as a throttle rather than a metric to be fine tuned.

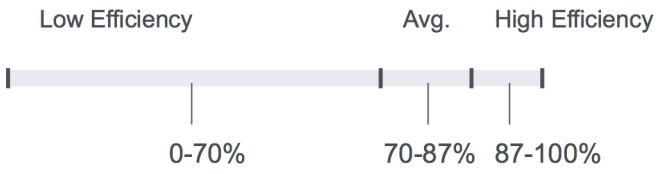

Figure 21: This scale provides insight into how to rate the efficiency metric for engineers.

7 Conclusion

In conclusion, this document provides supporting statistical evidence for GitPrime's benchmarking on Commits, Activity, Impact, and Efficiency. Our three major author segments are utilized in the Leadership Playbook and elsewhere in GitPrime's products. This broad understanding serves as a signpost for data driven leadership, which we promote through our own internal product development as much as we do in the delivery of our external services.