# Two Differing Approaches to Survival Analysis of Open Source Python Projects

Derek Robinson, Keanelek Enns, Neha Koulecar, Manish Sihag
*Department of Computer Science*
*University of Victoria*
Victoria, Canada
{drobinson, keanelekenns, nehakoulecar, manishsihag}@uvic.ca

## I. Motivation

The developers of Open Source Software (OSS) projects are often part of decentralized and geographically distributed teams of volunteers. As these developers are volunteering their free time to build software which is free for the masses they likely want to ensure the projects they work on do not become inactive. If OSS developers knew which projects would remain active and which would become inactive they could better decide if their free time was worth volunteering to a given project. Understanding which attributes of an OSS project lead to its longevity is what motived Ali *et al.* to apply survival analysis techniques commonly found in biostatistics to study the probability of survival for popular OSS Python projects [1]. We resonate with this motivation and would like to lend credibility to the findings of [1] by replicating the study. In addition to the replication study, we also plan on applying a Bayesian approach to survival analysis as outlined in [2]. The Bayesian portion of our paper is motivated by wanting to compare the findings of the two different approaches to survival analysis.

Thus, the research questions we plan on answering are as follows:

1) What attributes of an OSS project lead to its survival?
2) How do the findings of frequentist survival analysis differ from Bayesian survival analysis?

## II. Research Strategies

### A. Method

1) We will follow the methods outlined [1] for the replication portion of our study.
2) In the comparison portion of our study we will take a Bayesian approach to survival analysis as outlined in [2]

### B. Data

In order to perform survival analysis using the methods discussed above, a dataset which records the repositories for projects on common VCSs in their entirety, including, a history of all commits (revisions from here on out) and major releases (revisions of note, often with a specific name and release date) is required [1]. The *popular-3k-python* subset of the Software Heritage graph dataset [3] is what will be used in both our replication study and Bayesian survival analysis study. This dataset contains information on roughly 3000 popular Python projects which were hosted on Gitlab, GitHub, Debian, and PyPI between 2005 and 2018.

## III. Expected Results

We expect that our replication of [1] will yield similar results the original analysis. This is because we are attempting to follow the same approach in order to lend credibility to their findings. Additionally, as the survival analysis techniques outlined in [1] and the Bayesian approach to survival analysis outlined in [2] are two differing approaches to the same goal, we suspect that the results of both will be comparable.

## IV. Limitations

**[Derek: I feel like we need to brainstorm some limitations because I am at a loss]**

## References

[1] R. H. Ali, C. Parlett-Pelleriti, and E. Linstead, "Cheating death: A statistical survival analysis of publicly available python projects," in *Proceedings of the 17th International Conference on Mining Software Repositories*, 2020, pp. 6–10.

[2] R. Kelter, "Bayesian survival analysis in stan for improved measuring of uncertainty in parameter estimates," *Measurement: Interdisciplinary Research and Perspectives*, vol. 18, no. 2, pp. 101–109, 2020.

[3] A. Pietri, D. Spinellis, and S. Zacchiroli, "The software heritage graph dataset: public software development under one roof," in *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE, 2019, pp. 138–142.