# CSC 591: Automated Software Engineering | Project Report
# Improving Tested - An Amortized Analysis for Randomized Base Projections


Ankur Banerji
*Dept. of Computer Science*
*College of Engineering*
*North Carolina State University*
*NC, 27606*
*Email: abaner24@ncsu.edu*

Luke Jenquin
*Dept. of Computer Science*
*College of Engineering*
*North Carolina State University*
*NC, 27606*
*Email: lgjenqui@ncsu.edu*

Saurabh Nanda
*Dept. of Computer Science*
*College of Engineering*
*North Carolina State University*
*NC, 27606*
*Email: saurabhnanda.official@gmail.com*



***Abstract*—In this project, we aim to improve upon "Fishing", an active semi-supervised learning algorithm within the toolkit "TESTED" [1]. The primary objective is to address problems such as fairness, class imbalance and minority representation, when it comes to sampling in our stakeholder explanation system. Whether used as an acceptance testing toolkit for purposes such as QA or used as a Stakeholder Explanation System for getting a big picture to judge an AI system for important metrics such as fairness, any improvement upon our Fishing.**


## 1. Introduction

$TESTED^{[1]}, is \quad a \quad semi - supervised \quad , multi - objective \quad , model - based \quad explanation \quad system.$

TESTED is used as a stakeholder explanation system, to help stakeholders who are using an AI system to test it, and mitigate the issues associated with the AI based software. Stakeholder testing here, refers to testing the software by the stakeholders involved by trying to 'break' the system and using efficient and representative sampling methods, explore the system and fix its issues.

Stakeholders such as a person simply using an AI tool, need not have all the information regarding the system or how AI works, in general. This is where TESTED comes into picture, by helping users mitigate the problems associated with the AI system by getting a big-picture analysis and get a picture of the gradient of errors, rather than concrete answers and quick fixes.

The main algorithm or functionality that is the object of this project, is the 'Fishing' Algorithm, which is a semi-supervised active learning algorithm.

The constraint here is on the sampling budget, B0 (say). For a number of examples, N; as chosen by our algorithm, we need to ensure that

$$B0 << N$$

. This is ensured by our algorithm as classification would require us to build a binary tree that accesses the data a single time, and builds to a depth of log2(N) and in each of these steps; where the clustering requires accessing (N/2) examples twice, once for each cluster till the number of nodes is reduced to root(N/2). Recursively, adding up the number of steps, we get:

$$2^* \ [\sqrt{(N/2)} + \sqrt{(N/4)} + \sqrt{(N/8)} + ...\sqrt{(root(N))}]$$

This sums up to:

$$\sqrt{(n)} * (1 - (1/\sqrt{2}) * * \log_2 n)/(1 - 1/\sqrt{2})$$

Therefore, say for an n=1000, we should have a budget of about 210, which satisfies the condition.

### 1.1. Initial Introduction

**1.1.1. Everyone does Semi-Supervised Learning.** Semi-supervised learning is used when a small portion of the data is labelled while the other portion is un-labelled. An unsupervised learning method, in general, is drawing patterns from given data without having prior knowledge of any existing patterns, for ex., finding patterns in news articles without knowing if there is a pattern, using an unsupervised clustering method. As a practical application, this method is performed by Google news for a user's daily feed.

Semi-Supervised learning has many advantages in situations where obtaining "y" or "target" data may require human annotators, special devices, or expensive and slow experiments [4]. It is especially useful when the cost of sampling the target is high or when getting labels for a high volume data is not possible. For example, in a drug trial collecting the target variable for a large population based on several traits, would be quite expensive [1].

As a result, it is used widely in many applications to improve upon performance metrics, reduce sampling and

collection costs for the target variable, etc, usually in multiple runs or iterations, where progressively labelled data might be used to improve upon the classifying model.

**1.1.2. Problem with most Semi-Supervised Learning Methods.** The problem with most semi-supervised learning algorithms is associated with the sampling method not representing the entire population accurately. A form of this is the class imbalance problem. If the sampling is performed such that the minority class is not represented accurately, then there is a class imbalance problem, because the samples do not fairly represent the entire population.

Another relevant problem is the "novel class problem". In this, in semi-supervised learning models, it is quite common to encounter novel classes from an unlabelled pool. This may cause "unreliable sampling", as the researchers might require collection of this data for later use, and the classifier might label the data according to new classes, or try to fit new classes with the help of known data.

**1.1.3. Our Novel Insight.** The way clustering worked in TESTED was to take a population and divide it into half. One would then throw away the points that are least relevant, that is, beyond a certain confidence threshold (95%, for example). After dividing the data, the algorithm would perform the semi-supervised learning from the available data on the y labels and perform clustering till it hit the root(n) mark for each left and right cluster. For symmetry and the possibility of encountering better points on both sides of the cluster, one would merge these, and switch the left and right clusters. This, when performed recursively, would cluster the data into two sections, that is, the 'best points' and the 'rest points', where our focus lies on the 'best'.

We noticed that to perform this clustering, one would affix a point at the far edge of the cluster, based on it's cosine projection from the cluster's centroid, and this would be fed to the model as the reference point for labelling the data. The problem with this method is, in general, that for a given data, one can never be always certain that getting the farthest point as the baseline would yield the best outcome for clustering.

**1.1.4. What we did to solve the problem.** *Randomized Base for Projections- An Amortized Analysis*: To solve the problem mentioned in the previous section, instead of performing an exact analysis on the run time, one can focus on performing an amortized analysis [8].

Instead of choosing a far point as the the base for our cosine projections (that help label the data), we randomize the process of choosing the base instead, and calculate the amortized time. This enables us to look at the overall efficiency of the algorithm over many runs, instead of picking the farthest point and neglecting another point that might fare better as the base for projections, when it comes to clustering.

## 1.2. Structure of this paper

In this paper, we aim to propound upon ways to improve upon TESTED- A stakeholder explanation system. To do so, we start exploring the algorithm it is built on, vis-a-vis 'Fishing'.

'Fishing', is a semi-supervised learning algorithm, which uses 'SWAY' [13] under the hood. We start by providing a brief introduction of TESTED and the constraints we have on our sampling budget, the context of how our approach and proposed improvements are better than many common approaches for tackling the problem.

We explain our approach for reaching a randomized solution and how to perform analysis, and further discuss which form of an analysis is better suited for measuring the performance issues pertaining to the problem at hand (Section 1.1.4). We then explain the research questions we are trying to answer, the caveats attached with our proposed approach (stated implicitly or otherwise), and potential issues that might evolve from the same.

In the next section (Section 2), we touch upon Past Work and State of the Art Systems that our work is built on, influenced by or related to. In Section 3, we talk of the Methods and Technologies involved and description of the Data we are using to arrive at our conclusions. We also discuss performance and summarizing methods under this section, to describe what we are trying to accomplish.

In Section 4, we discuss our Results for the study. We discuss our Sampling Methods, results from our Prudence Study, and in the last section, that is, Section 5, we discuss the threats to validity of our model and the insights from the same. We also discuss our Future Scope and possible avenues for improvement.

The final Section, Section 6, provides our Conclusion for the same, followed by Acknowledgements and References we used for our work.

**1.2.1. List of Research Questions.**

**1.2.2. Overall Contributions of Team Members.** Contributions of members are as follows:

**Ankur Banerji**: Ideation phase for randomization, Ideation phase for Amortized Analysis, test cases, base code, etc.

**Luke Jenquin**: Ideation phase for randomization, Ideation phase for Amortized Analysis, Test Cases, etc

**Saurabh Nanda**: Ideation phase for randomization, Ideation phase for Amortized Analysis, Project Report generation, Code for several unit tests throughout project and homeworks it is built on, User Study and Analysis for Repgrids Sessions, Debugging, Citation Management, Proof Reading, etc.

**1.2.3. Caveats.** The possible caveats and future scope posed in our study is as follows:

- We have performed an amortized analysis, by randomizing the base for our projections, while performing sway and xpln on our datasets. This is based on the assumption that choosing the furthest point all the time might not be the best strategy for the classification. It is possible that this assumption is not always true, and hence we used amortized, instead of precise runtime. We are more concerned about how the algorithm performs on average, than if it performs the best in a certain situation.
  The caveat, of course, is that it is entirely possible that empirically, our algorithm does worse on datasets and our proposed method might suggest. In such a scenario, we need to inspect the cause of this deviation from our theoretical model and it might help us improve upon our model further in the future.

- The second caveat is that for many machine learning methods, type of data is also an important factor. Skewed data, data which is centered better on farther points, highly asymmetrical data in the two halves[], might be some of the concerns we have to deal with.

- Another caveat we face is that no process is truly random, or the overall issue of 'pseudo-randomness' of our generator. This issue is quite prevalent and finding an optimal generator too will affect our choice of a base point, and in turn, our fishing algorithm.

## 2. Related Work

### 2.1. Past Work

The past work involving semi-supervised learning tools like Fishing involve improving upon sampling methods [16], and often focus on not taking into account improving using an ad-hoc sampling. The ad-hoc nature of sampling is seen as a bias in sampling methods, and explaining the results often involves discussion over sampling bias and snowball affect.

### 2.2. Rationale for improving upon Past Work

### 2.3. Examples of State of Art Systems from Past Work

-An overview of related work, then breaking it down in 4 subsections

## 3. Methods

### 3.1. Algorithms

The primary algorithm used in TESTED toolkit is "Fishing", which is a stakeholder explanation system. A stakeholder can be anyone, such as business partners or a potential client. One does not need to know much about the AI system to perform the analysis. The tool should provide us information on whether our AI software is fair, performs well under constraints such as tax, and is overall suited for the application. Under the hood, we use semi-supervised clustering in a classifier tree. We use the algorithm SWAY to throw away the outliers (above 95 percent), cluster the data items and then merge, and repeat for it's mirror image (for non-symmetric data). This when broken down, is a combination of several important algorithms, including SWAY, Clustering, Halfs (for dividing the data cloud), Merge (to merge the data point cloud) and Minkowski Distance (p-distance, with default set to 2 or Euclidean). Finally, our optimization uses a random generator algorithm to introduce randomness in choosing the point that is different than the furthest point in the data cluster.

### 3.2. Data

- The AutoCSV dataset: It is used in predicting the attribute "mpg". The primary motivation when this dataset was first used, was to detect mpg. The data concerns city cycle fuel consumption in miles per gallon, to be predicted in terms of 5 continuous and 3 multi-valued discrete attributes, and 8 of these entries have values of "mpg" attributes removed because they had unknown values in original instances.

- China.csv China.csv is a part of COCOMO project for software effort estimation [10] ,where COCOMO stands for Boehm's COnstructive COst MOdel (COCOMO). It is a parametric estimation method, that is, it assumes given model has a particular structure and then uses model-based methods to fill in the details of that structure.

- Health Closeness CSV
  In this dataset, we are using an Extra Trees Classifier [13], which is a meta estimator, that is, it aims to improve performance by aggregating predictions of multiple estimators or models.

In this way, an ensemble method such as Extra Trees Classifier in sklearn, learns a meta estimator. This meta estimator, in our use case is the randomized decision trees (a.k.a. extra-trees) in our specific use case. Our objective is to minimize MRE (Magnetic Resonance Enterography), which simply put, is a type of MRI imaging method.
At the same time we want to maximize Agnesis of Corpus Collusum Callosum (ACC), which in short, is a brain disorder in which the tissue that connects the left and right sides of the brain (its hemispheres) is partially or completely missing; and its predictor PRED40. In a nutshell, we want to be able to predict the disorder ACC, and maximize it's corresponding predictor Pred40 with the minimum amount of scanning.

- POM CSV
  POM csv is the data file extracted from the POM3 project, which, simply put, is a software model for planning Agile development and related processes such as team based sprints. It can be visualized as a heap of intra-dependent requirements [11], such that for any requirement to be satisfied, it's children and dependencies have to be satisfied first. Each requirement has a cost and a value associated with it. One of the main problems associated with Agile is that of prioritization based on cost or salaries paid to the developers vs the number of hours put in by each developers. Other factors such as seniority of developers and criticality of the software engineers are also aspects that have to be taken into account. In our specific use case, our objective is to minimise on the attributes cost and idle time for the projects, while trying to reach maximum completion. Other variables like team "Size", Plans involved, Interdependencies, etc are used in the classification process as independent or treatmemnt variables.

- SSM CSV Most Software systems today are configurable, and keeping track of configurations becomes increasingly harder as the configuration space becomes larger. The SSM dataset is used in context of the Trimesh library; which is a library used to manipulate Triangle meshes [12]. The primary aim of our classification is to minimize on attributes "TimeToSolution" and "NumberIterations", which represents the time taken to reach a solution and the number of iterations involved for the same. Other treatment variables include, parameters like 'alpha' and 'beta', measures like 'jacobi' constant, and attributes like 'colorsGS', etc that represent the properties of the triangles.

- SSN CSV Similar to SSM dataset above, from [12], the context is to keep track of software configurations as the the configuration space becomes larger. The SSN dataset deals with the configuration space of an X-264 video encoder. Our main objective with this dataset is to minimize PSNR (Peak Signal to Noise Ratio) and Energy attributes. Independent attributes include variables like seek in tape, threads, lookAhead, Bframes, etc.

### 3.3. Performance Measures

Expected Results: From the tree analysis, we expected two things:

- Reduction in tree height: We expected fewer calls on the stack for clustering along our classifier tree. This is because, even if random point in cluster as the base is not the best point all the time, randomizing at least half the time should produce better amortized results for the entire run.
- Better performance in terms of leaf nodes, most of the time, if not all the time. This is because randomisation should improve upon instances where farthest point is not the best choice for projection, at least some of the time.

### 3.4. Summarization Methods and Results

- Experimental Reduction in tree height: The tree height remains almost the same, but reduces by a level. An actual prune for the entire subtree was rare. This is somewhat optimistic but not entirely as expected.

- Experimental performance in terms of leaf nodes was expected to improve. It was observed that when it improves, it improves as much by a scale of about ten times. However, in a few leaf nodes, the results were also as worse as ten times which has a room for improvement. Even if our algorithm can do good on average, and not for all leaf nodes, we can be optimistic of it as a heuristic measure, if not a concrete optimization.

## 4. Results

### 4.1. Run with Sampling Methods

First off, we observed that randomization did not affect the sampling tax. This is as expected, as we do not implement a method that would drastically change the depth of the tree. Our Sway optimization, in the worst case, could potentially build a linear tree of length n. However, this case is extremely rare, for two reasons:

- Firstly, we do not always use the random generator, but only for a fraction of the time. In our sway implementation, we do not throw away the farthest projection, but simply add an element of

randomness. Even if the random generator does the worst, some of the time, for some samples, it does not work all the time and it can't do worst all the time, by the definition of it being a random process.

- Secondly, as observed empirically, random generator does do better sometimes upto 10 times for a leaf node, so it is extremely rare for it do worst, without getting pruned and simply resulting in lengthening of the tree depth.

### 4.2. Discussion of Results

Randomization does not affect sampling tax. Discuss Trees and improvements.

## 5. Discussion

### 5.1. Threats to Validity

As discussed in Section 1.2.3, the biggest threat to our validity would be:

- The veracity of our random generator and the concept of pseudo-randomness. For a good random generator, we might expect better results, but defining what constitutes as 'good' or more random, is also a challenging feat.
- Once we do agree on a random generator, the true test would be to test it on harder datasets. By harder, in this context, one would mean intentionally skewed datasets tilted towards one side, with base projection not affixed at far points.

### 5.2. Big Picture Insights

By moving towards more randomized sampling, and focussing on explaining ad-hoc results rather than grouping it all under bias [16], we can draw several meaningful insights. It is useful to note that one only need to perform better most of the times, which is optimistic and is seen in our case study on analyzing the tree classifier in action.

### 5.3. Future Work

- With reference to the second point in Section 1.2.3 above, we see that our method might fail in some instances whether data is highly asymmetric and dividing it in half might no solve the problem. We aim to improve upon this by dividing it in other ratios, and aim to find out an optimal ratio which could help solve this problem of asymetric-ity.

- With reference to the final point in section 1.2.3, we face the issue of finding a good random generator that is truly random or more truly random. This would involve experimenting with a lot of choices or making our own, which is quite a pertinent topic for our future scope.

## 6. Conclusion

Our objective for the study to improve upon the Fishing algorithm, in the tookit TESTED was two-folds:

- To introduce randomness in SWAY, so as to not get biased projections from the farthest point. We succeeded in introducing randomness, but instead of always going randomly, we chose a hybrid approach, deciding to randomize for only a given fraction of time.
- To improve upon the sampling method by improving on SWAY and as a result, improve upon explaining the results. We hoped to reduce the length of the classifier tree, and thus improve upon the overall efficiency, some of the time, if not all. We therefore performed amoritized analysis for the same. However, we noticed that for most cases, since the depth of the tree remains the same or close to our original analysis, and some cases perform as badly as ten times, while others perform as good as ten times, further analysis is required. For example, one needs to test the classifier for highly skewed data, run it for several runs and check the prune frequency of the tree. If the prunes are more common and the average of ten times for the leaf output can be further maximized, we could further improve upon our projected outcomes. As such, there is room for improvement, and we leave it as part of the future scope.

## Acknowledgments

The authors would like to thank...

- Prof. Tim Menzies, CSC 591 ASE, NCSU
- Rahul Yedia, Teaching Assistant, CSC 591
- Andre Motta, Teaching Assistant, CSC 591
- Xueqi Yang, Teaching Assistant, CSC 591

## References

[1] Tested, Tim Menzies, BSD-2-Clause licence.

[2] B. Settles, "Active learning literature survey," MINDS@UW Home, 01-Jan-1970.

[3] Yang, Yuzhe and Xu, Zhi, "Rethinking the Value of Labels for Improving Class-Imbalanced Learning", 2020

[4] J. Brownlee, "What is semi-supervised learning,", 16-Dec-2020. [Online].

[5] Helman, D. H., & Bahuguna, A. (1986). Explanation Systems for computer simulations. Proceedings of the 18th Conference on Winter Simulation - WSC '86. https://doi.org/10.1145/318242.318476

[6] Dazeley, R., Park, S. S., amp; Kang, B. H. (2011). Online knowledge validation with Prudence Analysis in a document management application. Expert Systems with Applications, 38(9), 10959–10965. https://doi.org/10.1016/j.eswa.2011.02.139

[7] Dazeley, R., Park, S. S., amp; Kang, B. H. (2011). Online knowledge validation with Prudence Analysis in a document management application. Expert Systems with Applications, 38(9), 10959–10965. https://doi.org/10.1016/j.eswa.2011.02.139

[8] Probabilistic analysis and Randomized Quicksort. (n.d.). Retrieved April 20, 2023, from https://www.cs.cmu.edu/afs/cs/academic/class/15451-s07/www/lecture_notes/lect0123.pdf

[9] Burgner-Kahrs, Jessica . “Amortized Analysis, Quicksort & Randomized Algorithms.” University of Toronton.d. https://mcs.utm.utoronto.ca/~263/jessica/lecture6.pdf.

[10] Tim Menzies, Ye Yang, George Mathew, Barry Boehm, and Jairus Hihn. 2017. Negative results for software effort estimation. Empirical Softw. Engg. 22, 5 (October 2017), 2658–2683. https://doi.org/10.1007/s10664-016-9472-2

[11] IEEE. (2018, January 5). “Sampling” as a Baseline Optimizer for Search-based Software Engineering. arxiv. Retrieved April 21, 2022, from https://arxiv.org/pdf/1608.07617.pdf.

[12] Nair, V., Apel, S., Siegmund, N., Menzies, T., amp; Yu, Z. (2018). Finding Faster Configurations Using FLASH. https://doi.org/https://arxiv.org/pdf/1801.02175.pdf#page=2

[13] Chen, J. (2018, January). “sampling” as a baseline optimizer for search-based software engineering. Research Gate. Retrieved April 21, 2023, from https://www.researchgate.net/publication/322324736_Sampling_as_a_Baseline_Optimizer_for_Search-Based_Software_Engineering

[14] Sklearn.ensemble.extratreesclassifier. scikit. (2023). Retrieved April 21, 2023, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html

[15] U.S. Department of Health and Human Services. (n.d.). Agenesis of the Corpus Callosum. National Institute of Neurological Disorders and Stroke. Retrieved April 21, 2023, from https://www.ninds.nih.gov/health-information/disorders/agenesis-corpus-callosum

[16] Baltes, S., amp; Ralph, P. (2022, April 28). Sampling in software engineering research: A critical review and guidelines - empirical software engineering. SpringerLink. Retrieved April 21, 2023, from https://link.springer.com/article/10.1007/s10664-021-10072-8