Team 17
A multi-objective semi-supervised explanation system using Mini-Batch KMeans

Prachit Sandesh Mhalgi
Department of Computer Science
North Carolina State University
Raleigh, NC
Email: psmhalgi@ncsu.edu

Sahil Santosh Sawant
Department of Computer Science
North Carolina State University
Raleigh, NC
Email: ssawant2@ncsu.edu

Neha Vijay Patil
Department of Computer Science
North Carolina State University
Raleigh, NC
Email: npatil2@ncsu.edu

I. Introduction

We know that the datasets are varying and there is no proven single optimized searching way, although there is a currently used industry-standard method called Sway, also short for Sampling Way, which works on this idea and tried to find the close to the optimum solution while reducing the computational cost. The aim of this project is to make a multi-objective semi-supervised explanation system. The previous system used the Fastmap algorithm for clustering and SWAY as an optimizer. The Fastmap algorithm is only one of the many options for clustering algorithms. Although it shows its efficiency for smaller datasets, it can be understood that as the size of the sample increases it starts performing worse, to an extent that the compromise on the computational efficiency may not work as a justifiable measure. FastMap is a simple algorithm that does heavy dimensionality reduction. Therefore, the experiment is to to research over other clustering algorithm and utilize them in a way that the increasing sampling size does not affect the performance of the algorithm. In this experiment, we will specifically be working on the K-Means algorithm

A. Structure of this paper

- We have first done a literature survey on the existing system and the clustering algorithms.
- Explored the continuous domination method.
- Explained the Mini-Batch KMeans algorithm.
- Description of datasets.
- Explained the different statistical methods, justification of our choice and their results.
- Conclusion and future work

II. Related Work

As the state of art system for multi-objective semi-supervised explanation system, we consider AI, tested system which was studied in class. We will compare our results to this system. The method followed in class was based on the novel algorithm- SWAY described in [2]
Firstly, they used FASTMAP to cluster the points. It is a simple and fast PCA based algorithm which simplifies the clustering process to near linear time. Two distant points east and west are chosen. Distances of these are calculated as an arbitrary point C. Assuming N dimensions of data, we project them to a single dimension which is the line connecting east and west. These points are then assigned to closest cluster point among east and west according to their distances.
This FASTMAP algorithm is utilized in SWAY technique. FastMap returns the data divided in two clusters. Sway is an optimization algorithm that looks at Y values to determine the better cluster. The better cluster is identified using continuous domination (zitzler's predicate). The worse half is discarded and the better half is recursively divided again.
Continuous domination determines how to determine better between competing concerns. If you have more than 3 goals, zitzler's predicate is used to decide [9]. For each goal, we calculate the cost of forward and backward jump. This cost is based on whether we want to maximise/minimize the goal, values of the goal and total number of goals. Formula for forward(s1) and backward(s2) jumps is given:

$$s1 = -\sum e^{wi(ai-bi)/n)}$$

$$s2 = -\sum e^{wi(bi-ai)/n)}$$

one example is better than another if we lost the least jumping to it; i.e. $s1 < s2$
The issue of fastmap is although it is super fast, it is also very simplistic. It assumes two clusters everytime and SWAY drops the worse half. The assumption of 2 initial clusters might result in dropping of many good examples. Additionally, converting N dimensions to 1 dimension results in loss of data that could have significant meaning.
Mini Batch K-means is also a linear time , non-supervised and easy to implement clustering algorithm. It allows us the flexibility to choose more than two initial clusters which might result in better results. Also, it does not do dimensionality reduction. [8]

III. METHODS

A. Algorithm

Clustering is the process of grouping similar examples together. For our purposes, clustering will be applied on X attributes only, because evaluating for Y values is very expensive. Clustering enables us to approximate large amounts of data by the clusters' centroids. This reduces the redundancy in data and increases speed in processes such as requirements engineering and labeling problem.

K-means is a simple unsupervised clustering algorithm. We start with k random centroids and label every other point with their closest centroid. After each iteration, we move the centroids to the mean value of their respective cluster. This process is applied again until there is no change to the centroids.[3]

For large datasets like ours, K-means has a large computational cost, as it calculates distance for all data points till convergence. Mini-Batch Kmeans however, at every iteration it only does distance computations for a pre-defined batch size of say 500. This significantly reduces the computational cost of clustering in massive datasets.[1]

For this paper, the hyperparameters chosen were batch size=250 and number of clusters=2.

This is how the algorithm works. It first randomly chooses two (number of clusters) examples as centroids. Then, next 250 (batch size) examples are labeled by their closest centroid. Before running it for the next batch, we move the centroids towards their respective cluster. Each example "pulls" its centroid "c" towards its own attribute "x" by an amount weighted as:

$$c = (1 - 1/n) * c + x/n$$

where n is the number of times "c" has been chosen by examples.

In our usecase, we have defined two datatypes for attributes: NUM(numeric) and SYM(symbolic). For NUMS, the data is described using mean(central tendency) and standard deviation(diversity). The distances among NUMS attribute is calculated using L2 norm (Euclidean distance) on normalized data. For SYMs, the data being categorical in nature, is described using mode(central tendency) and entropy(diversity). The distances among SYM attribute is 1 for unequal symbols and 0 for same symbols. For our data, the overall distance metric used is called AHA and is explained below:

If NUMS: Euclidean distance

If SYM: return 1 if x!=y

For missing values: assume worst case

- if both unknown, assume distance = 1
- if one symbol missing, assume distance= 1
- if one number missing:
  - let x,y be unknown, known
  - y = normalize(y)
  - x = 0 if y ¿ 0.5 else 1
  - distance = (x-y)

Dataset name	**Description**	**Goals (+: maximise, -: minimize)**
auto93	Car properties	Lbs- Acc+ Mpg+
Auto2	Car properties (More detailed)	CityMPG+ HighwayMPG+ Class-
China	SE project effort estimation.	N_effort-
Coc1000, coc10000	SE project effort estimation using COCOMO parameters.	LOC+ (lines of code), AEXP- (application exp), PLEX- (platform exp), RISK-, EFFORT-
Nasadem	SE project effort estimation	Kloc+ Effort- Defects- Months-
health-hard.csv health-easy.csv	Hyperparameter optimization of extra trees classifier	MRE- (error), ACC+(accuracy), PRED40+
pom	Agile project management	Completion+, idle- cost-
SSM	Configuration space for manipulating triangle meshes	Iterations-, Time to solution-
SSN	Configuration space of X-264 a video encoder	PSNR-, Energy-

TABLE I: Data Description

B. Data

We have 11 datasets to explore. Table 1 shows each dataset's description. We have not mentioned the many X attributes that are present due to space constraints. In a multi-objective system we have multiple goals to either maximize or minimize. These goals for each dataset are listed in Table 1.

NUM attributes can be described using minimum value(low), maximum value(high), mean and standard deviation. SYM attributes are described using mode (central tendency) and entropy (diversity). The same is shown for two of the eleven datasets in table 2 and table 3.

type	name	low	high	mid	div
NUM	Clndrs	3	8	5.454773869	1.701004245
NUM	Volume	68	455	193.4258794	104.2698382
NUM	Model	70	82	76.01005025	3.697626647
SYM	origin			1	1.327355848

TABLE II: Attribute summary for auto93.csv

	Lbs-	Acc+	Mpg+	n_evals avg
all	2970.42	15.57	23.84	0
sway1	2180.18	16.9835	32.249	6
sway2 (mini batch kmeans)	2379.46	16.92	27.69	5
xpln1	2244.26	16.6865	29.621	6
top	1991.91	19.52	40.91	398

Fig. 1: Results of SWAY and Modified SWAY on Dataset Auto93

	Lbs-	Acc+	Mpg+
all to all	≠	≠	≠
all to sway1	≠	≠	≠
sway1 to sway2	≠	≠	≠
sway1 to xpln1	≠	≠	≠
sway1 to top	≠	≠	≠

Fig. 2: Results of SWAY and Modified SWAY on Dataset Auto93

C. Summarization Methods

We want to now compare results from different fishing methods. This is achieved using statistical techniques. There are two types of statistical tests:

TYPE	name	low	high	mid	div
NUM	AFP	9	17518	486.8577154	1059.171436
NUM	Input	0	9404	167.0981964	486.3385747
NUM	Output	0	2455	113.6012024	221.2743742
NUM	Enquiry	0	952	61.6012024	105.4228399
NUM	File	0	2955	91.23446894	210.2709839
NUM	Interface	0	1572	24.23446894	85.0409961
NUM	Added	0	13580	360.3547094	829.842333
NUM	Changed	0	5193	85.06212425	290.8570395
NUM	Deleted	0	2657	12.35270541	124.2241296
NUM	PDR_AFP	0.3	83.8	11.77054108	12.10564913
NUM	PDR_UFP	0.3	96.6	12.07975952	12.81871009
NUM	NPDR_AFP	0.4	101	13.26973948	14.00983971
NUM	NPDU_UFP	0.4	108.3	13.62625251	14.84341606
NUM	Resource	1	4	1.458917836	0.823729173
NUM	DevType	0	0	0	0
NUM	Duration	1	84	8.719238477	7.347058438

TABLE III: Attribute summary for china.csv

- Effect-size test tells us if the results from the methods are different by more than a trivial amount.
- Significance test tells us how much overlap is there between the results of two methods. It helps us determine if they're from the same population distribution or not.

These tests can also be of two types:

- Parametric: Here, we have the information about the population. Usually, it is assumed to be a normal distribution. While such an assumption would be untrue, it simplifies the analysis.
- Non-parametric: No assumptions are made about the parent population and the tests are applied solely to the sample data.

	N_effort-	n_evals avg
all	4277.64	0
sway1	3698.79	6
sway2 (mini batch kmeans)	167.31	7
xpln1	3430.73	6
top	152.233	499

Fig. 3: Results of SWAY and Modified SWAY on Dataset china.csv

	N_effort-
all to all	≠
all to sway1	≠
sway1 to sway2	≠
sway1 to xpln1	≠
sway1 to top	≠

Fig. 4: Results of SWAY and Modified SWAY on Dataset china.csv

The point of these tests is to make sure our results have any concrete inferences. If the two methods are significantly different(significance test) and they are different by more than a trivial amount, only then we can make any comparisons between the two methods. Otherwise, we can make no conclusions.

For effect-size testing, we have chosen to do a non-parametric test to avoid the assumption of normality. [6] One such method is the cliff's Delta effect size test. [7][5]

Cliff's delta values are interpreted as below:

- $d < 0.147$: negligible
- $d < 0.33$: small
- $d < 0.474$: medium
- $d > 0.474$: large

For significance testing, we have again chosen to do a non-parametric test to avoid the normality assumption. One such method is the bootstrap method.

This is how bootstrapping works:

- Make bootstrapped datasets using random sampling with replacement.
- Calculate the means of the datasets so you have a distribution of these bootstrapped datasets' means.
- if the mean of a bootstrapped sample is different from the sample, then we count that. we use this to report if the distributions are different. [4]

IV. Results

A. Prudence Study

Initially, we started using K-means as a clustering method in order to perform sampling but from the experiments, as the sampling size increased, the computational cost and computational time for the same increased drastically as well. Thus we switched over to minibatch K-means, in which we selected the hyperparameter of the batch size to be 250, this helped in reducing the computational cost significantly as it only considers sampling sizes in batches as defined in the hyperparameter. Thus, we do not computer the entire sample space for every single iteration, but proceed to do it in smaller batches. Thus, it improvises and adapts well to an increasing sampling size

	Cost-	Completion+	Idle-	n_evals avg
all	369.99	0.87	0.24	0
sway1	247.311	0.87	0.2275	8
sway2 (mini batch kmeans)	425.23	0.84	0.18	7
xpln1	276.475	0.8665	0.245	8
top	150.08	0.96	0	10000

Fig. 5: Results of SWAY and Modified SWAY on Dataset pom

	Cost-	Completion+	Idle-
all to all	≠	≠	≠
all to sway1	≠	≠	≠
sway1 to sway2	≠	≠	≠
sway1 to xpln1	≠	≠	≠
sway1 to top	≠	≠	≠

Fig. 6: Results of SWAY and Modified SWAY on Dataset pom

B. Discussion

From the experiments, we derived a few quantifiable conclusions which emerged to be topics for discussion. Although we believed that minibatch k-means would be a better clustering alternative to improvise, over the current Fastmap algorithm implemented, it actually seemed to perform otherwise. In the beginning, when tested on the auto93.csv dataset[], we noticed that the results were actually better for the original sway algorithm, but as we proceeded ahead with the experimentation, we noticed that minibatch k-means performed better compared to the Fastmap algorithm. It is most significantly evident in datasets like coc1000.csv[], coc10000[], SSN.csv[].

	Loc+	Risk-	Effort-
all to all	≠	≠	≠
all to sway1	≠	≠	≠
sway1 to sway2	≠	≠	≠
sway1 to xpln1	≠	≠	≠
sway1 to top	≠	≠	≠

Fig. 7: Results of SWAY and Modified SWAY on coc1000 Dataset

But we also noticed that there were some parts of it that gave unexpected results, for example, in coc1000.csv[], we noticed

	LOC+	AEXP-	PLEX-	RISK-	EFFORT-
all to all	≠	≠	≠	≠	≠
all to sway1	≠	≠	≠	≠	≠
sway1 to sway2	≠	≠	≠	≠	≠
sway1 to xpln1	≠	≠	≠	≠	≠
sway1 to top	≠	≠	≠	≠	≠

Fig. 8: Results of SWAY and Modified SWAY on coc1000 Dataset

that even though most results were better for minibatch k-means, a particular column, "Risk-" delivered a better result for the Fastmap algorithm.

	Loc+	Risk-	Effort-
all to all	≠	≠	≠
all to sway1	≠	≠	≠
sway1 to sway2	≠	≠	≠
sway1 to xpln1	≠	≠	≠
sway1 to top	≠	≠	≠

Fig. 9: Results of SWAY and Modified SWAY on coc10000 Dataset

	Loc+	Risk-	Effort-
all to all	≠	≠	≠
all to sway1	≠	≠	≠
sway1 to sway2	≠	≠	≠
sway1 to xpln1	≠	≠	≠
sway1 to top	≠	≠	≠

Fig. 10: Results of SWAY and Modified SWAY on coc10000 Dataset

	PSNR-	Energy-	n_evals avg
all	44.53	1658	0
sway1	44.5515	1373.68	9
sway2 (mini batch kmeans)	36.55	303.23	9
xpln1	44.5185	1418.73	9
top	25.9815	503.928	53662

Fig. 11: Results of SWAY and Modified SWAY on SSN Dataset

Therefore we concluded the discussion with that, the mini-batch k-means has its own caveats, especially while dealing with smaller sampling sizes. Though, it could be safely said that it performs notably better as the sampling size increases.

V. Discussion

A. Threats to Validity

	PSNR-	Energy-
all to all	≠	≠
all to sway1	≠	≠
sway1 to sway2	≠	≠
sway1 to xpln1	≠	≠
sway1 to top	≠	≠

Fig. 12: Results of SWAY and Modified SWAY on SSN Dataset

- *As the proposed algorithm showed that the results were improving as the size of the dataset increased, while the computational cost was not affected by it. It can be validated that the increasing sampling size will not be a threat to the proposed algorithm*
- *The model has shown few abnormalities throughout the experiment when it performed marginally worse than the half algorithm for a few of the many columns in the datasets. Although, it was also observed that the same issue arises with the current algorithm. The cause of it couldn't be derived due to the limited extent of this experiment and is declared as an abnormal unexplained behavior*

B. Bigger Picture Insights

C. Future Work

- What sway is doing right now is throwing away worse of the two clusters created. The assumption of data being distributed in two clusters each time might result in the discarding of valuable examples. Thus, starting with more clusters might give us better results.
- Including simulated annealing or genetic algorithms to reach the goals better than the greedy SWAY method.
- Exploring more discretization methods and a better score function to choose the rules that have a better predictive power of good examples.

VI. Conclusion

In our system we used Mini-Batch Kmeans algorithm for clustering. We did this to avoid the heavy dimensionality reduction in Fastmap. They are both linear time clustering algorithms. Mini-Batch Kmeans gave us much better results than Kmeans for larger datasets. We analysed these results using statistical techniques : Cliffs Delta and Bootstrapping. We aim to pursue the points mentioned in future work to further enhance our system.

VII. B3. Ablation Study

For the Ablation Study, we compare mainly 3 algorithms, namely, Fastmap, Agglomerative Clustering and K-Means Clustering method. Upon comparing the results of the three, K-Means seemed to be a better performer over Agglomerative, although it did not fair well as the sampling size was increased. Thus we introduced an improvisation on K-Means by comparing it with minibatch K-Mean, ensuring that the overwhelming

sampling sizes would not affect the performance. Here, we conluded that although the differences between Fastmap and minibatch K-means are not noticeable on smaller datasets, as the size increases it becomes more evident.

VIII. B4. HPO Study

For the Hyperparameter Optimization Study, we experimented with the minibatch size, in order to find the most optimum set which would perform equally likely on smaller as well as larger sampling sizes. From the experiment, we concluded that the results stayed better, till the minibatch size was 40% of the sampling size and deteriorated after

References

[1] Javier Béjar Alonso. K-means vs mini batch k-means: a comparison. 2013.

[2] Jianfeng Chen, Vivek Nair, Rahul Krishna, and Tim Menzies. "sampling" as a baseline optimizer for search-based software engineering. *IEEE Transactions on Software Engineering*, 45(6):597–614, 2018.

[3] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

[4] David V Hinkley. Bootstrap methods. *Journal of the Royal Statistical Society: Series B (Methodological)*, 50(3):321–337, 1988.

[5] Guillermo Macbeth, Eugenia Razumiejczyk, and Rubén Daniel Ledesma. Cliff's delta calculator: A non-parametric effect size program for two groups of observations. *Universitas Psychologica*, 10(2):545–555, 2011.

[6] Jeanine Romano, Jeffrey D Kromrey, Jesse Coraggio, and Jeff Skowronek. Should we really be using t-test and cohen'sd for evaluating group differences on the nsse and other surveys. In *Annual meeting of the Florida association of institutional research*, 2006.

[7] Shlomo S Sawilowsky. New effect size rules of thumb. *Journal of modern applied statistical methods*, 8(2):26, 2009.

[8] David Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178, 2010.

[9] Eckart Zitzler, Simon Künzli, et al. Indicator-based selection in multiobjective search. In *PPSN*, volume 4, pages 832–842. Springer, 2004.