Uncovering User Information from Obfuscated K-Clustering

Malachi Coleman, Benjamin Eyre, Manuel Paez

December 13, 2024

Abstract

Large companies have access to enormous datasets, allowing them to derive insight about their users. Clustering algorithms like k-means can be used to group similar items together and discover commonalities. Companies may release anonymized versions of outputs from a clustering algorithm, but private data may still be vulnerable to leakage. As large corporations accumulate and analyze expansive user datasets, data privacy risks become increasingly complex. Even when sensitive information is not explicitly revealed, the outputs of unsupervised learning algorithms, such as the centers returned by a k-means clustering procedure, can provide adversaries with indirect insights into underlying user data. The release of obfuscated or anonymized cluster centers may thus inadvertently enable inference attacks that partially reconstruct private information. In this work, we present a concrete framework for a data-inference attack on k-means clustering, set within a performative prediction context. Our approach simulates an iterative, real-world scenario in which a company continually updates its model based on an updated dataset. During this iterative process, an adversary exploits this retraining and introduces known samples into the training data. The anonymized output of the online k-means algorithm yields labels for these samples that can be used to train a classifier that to approximates clusters learned from the original dataset. Finally, we analyze the conditions under which these attacks are effective and quantify the extent of the resulting privacy leakage. Our results demonstrate that even under this highly restricted threat model, an adversary can effectively extract information thought to be inaccessible. We release our code at https://github.com/calendarisle/Unsupervised_Machine_Learning_Fall24.

1 Introduction

Large companies leverage the vast size of their consumer base and subsequently, the data that is available to them, to create massive, diverse datasets fit for training machine learning models and data mining algorithms. These machine learning models and data mining algorithms in turn improve their services, increase their consumer base, and thereby increase the amount of data available to the company. This cyclical process, where the ability for predictions to influence their target, has been investigated in the supervised learning setting in the performative prediction literature of supervised learning. However, the connection to unsupervised learning is not well studied, which has a vast assortment of problems that large companies desire to solve. The k-means clustering problem, one of the most fundamental and widely studied problems in unsupervised machine learning, is such a problem that solving it can allow companies to group together data efficiently, and thereby extract valuable and concise information from massive datasets.

Machine learning and data mining algorithms are also prone to the leakage of confidential information about individuals who contribute data points. In certain scenarios, this can lead to

severe consequences, such as the loss of billions of dollars or IP theft. Thus, providing accurate algorithms and machine learning models that protect data privacy has become crucial in algorithmic and machine learning design. Given the importance of k-means clustering and preserving data privacy, interest in designing private clustering algorithms in Euclidean space and stopping data inference attacks against clustering algorithms have been investigated to a large extent. However, the cyclical process of performative prediction and its connection to privacy-based analysis and the unsupervised learning setting has not been well studied.

In this paper, we construct a concrete formulation of a data-inference attack on k-means clustering in an online, where the attacker gradually adds training data in order to create a labeled training dataset. In particular, given a dataset X and a k-means algorithm that takes in as input X and outputs the k centers, the user's (adversary) goal is to learn which regions of the input space are assigned to each cluster given the bounds of the input data X and the cluster labels for each example. We later relate this to the performative prediction literature by having the k-means labels influence where the adversary next queries. The main questions we study in this paper are the following:

1. Can we extract the identity of each cluster without knowledge of the underlying data?

2. How does a sampling procedure affect the efficacy of this attack and the noise on the labels?

One hypothetical application of our technique and real-world instance of a tech company releasing anonymized cluster information is Spotify Wrapped [1], where the music streaming giant assigns genre-based labels to users based on their listening history. While the exact determiners of these labels are not specified, knowing them might allow for adversaries to determine more about Spotify's customers than Spotify intended. Our approach describes how one could create a series of users to probe at this problem and uncover this information.

Ultimately, we are able to demonstrate that even naive query adversarial strategies are able to de-anonymize the cluster labels. We finally demonstrate that careful planning in the adversarial querying procedure can yield improved performance in certain regimes.

1.1 Related Work

Active Learning: The literature on Active Learning (also called "query learning") describes a regime the learning algorithm is allowed to choose the data from which it learns. Particularly, active learning algorithms attempt to overcome problems in the labeling by asking queries in the form of unlabelled instances to be labeled by an oracle, thereby hoping to achieve high accuracy with fewer labeled training instances. Several active learning algorithms have been proposed in frameworks that incorporate clustering [22], [18], [25], [6], taking into account prior data distributions and different sampling heuristics. Our work has a strong connection to this literature. However, our setting differs from the traditional active learning setting as in our setting, the learning algorithm is an adversary of the party receiving the unlabelled query examples.

Adversarial Algorithms: Adversarial algorithms for both online and offline k-means have been proposed. Zhang et al. [27] propose a general framework for online data poisoning attacks and apply this to k-means to the induce desired outcome for the learning algorithm. However, these authors assume that the algorithm will only be updated with their adversarial query data, while we assume an unknown 'unpoisoned' dataset will simultaneously be trained on. Huang et al. [14]

[1] https://www.spotify.com/ca-en/wrapped/

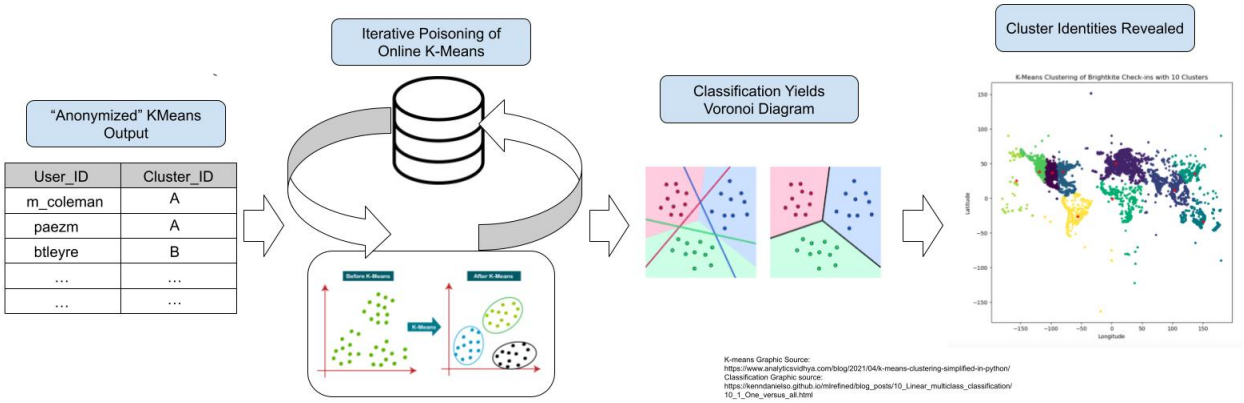


Figure 1: An outline of our general approach to this problem. 1. K-means training data is not directly released, but **cluster labels** are visible to the attacker. 2. The attacker iteratively adds data to the online k-means algorithm's training data, supplying **training labels** but gradually **shifting the centers**. 3. Once enough data has been extracted, a **multi-class classifier** is trained to approximate the associated Voronoi Diagram. 4. Approximating the hidden clusters effectively leaks insights about **users belonging to each cluster**.

presents an attack on online k-means that intends to infer the underlying training data. This attack assumes some knowledge of the underlying k-means algorithm. In contrast, the only knowledge our algorithm has about the target k-means instance is the number of clusters and the cluster labels on query points. Consequently, our threat model is far more pertinent to real-world scenarios, as we assume an extremely limited amount of information and influence on the target algorithm.

Performative Prediction: Our work relates to the performative prediction literature in supervised learning, where classifier predictions can affect training distributions in an online setting [20]. Other works have investigated how users can band together to influence the behavior of an online model [9]. Our work draws on the performative prediction literature through our *Stable Sampling* heuristic, which uses the labels produced by the online k-means target to help shape the distribution of future queries [9], [14].

k-**Clustering:** k-Means clustering has seen a large body of work that has spanned several decades. While the problem is known to be NP-hard [17], there has been a significant amount of work on various $O(1)$-approximation algorithms for the problem [2], [8]. Differentially private and other privacy-based k-means algorithms supply a probabilistic guarantee for the privacy of individual users [21], [24], [12]. Online and Streaming algorithms for k-means calculate a set of k-centers over time steps, [7], [16], [5], [1]. A supervised learning approach to k-means clustering [13] that learns a distance measure so that k-means produces the desired clustering has also been investigated.

2 Preliminaries

2.1 K-Means Clustering

We define $d(x, y)$ to be the Euclidean distance between two points x and y and for a finite subset $C \subset \mathbb{R}^d$, we define $d(x, C) = d(C, x)$ to be the $\min_{c \in C} d(x, c)$. given a dataset $X = \{x_1, ..., x_n\}$ of

points in $\mathbb{R}^d$ and a set of centers $C = \{c_1, c_2, ..., c_k\}$, we define the k-means cost as

$$cost(X; C) = \sum_{x \in X} d(x, C)^2$$

The goal in k-means is to find a subset C of k points that minimizes $cost(X; C)$.

2.2 Multi-Class Classification

Multi-class (also known as multi-label) classification is the learning task of learning to classify an object into one of many candidate classes. For our problem, we consider this multi-classification problem: for a distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the instance space and $\mathcal{Y} = \{1, ..., k\}$ is the label space. For multi-class classification, $|\mathcal{Y}| = k > 2$ (binary when $k = 2$). For $y \in \mathcal{Y}$, we want to learn a model that outputs a single class label y given an example $x \in \mathcal{X}$. A classifier (or hypothesis) is a map $h : \mathcal{X} \to \mathcal{Y}$, and the hypothesis class $\mathcal{H}$ is the set of possible multi-class classifiers $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$. The error of a classifier $h \in \mathcal{H}$ with respect to $\mathcal{D}$ is $Err(h) = Err_{\mathcal{D}}(h) = Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y]$. In the PAC setting, a learning algorithm for a class $\mathcal{H}$ is a function $A : \cup_{n=0}^{\infty}(\mathcal{X} \times \mathcal{Y})^n \to \mathcal{H}$. Given a training sequence $S_m = \{(x_1, y_1), ..., (x_m, y_m)\}$, an Empirical Risk Minimization (ERM) learner for the class $\mathcal{H}$ is a learning algorithm which takes in as input S_m and returns a classifier h where:

$$Err_{S_n}(A(S_m)) = \min_{h \in \mathcal{H}} \frac{1}{m} |\{i \in [m] : h(x_i) \neq y_i\}|$$

We can denote $Err_{S_m}(h) = \frac{1}{m}|\{i \in [m] : h(x_i) \neq y_i\}|$.

3 Technical Overview

In this section, we formally define the problem, the sampling heuristics, and querying models we experiment on and analyze. Let us define the setting of the problem: consider a dataset of n points $X = \{x_1, ..., x_n\} \subset \mathbb{R}^d$. A k-means euclidean distance clustering algorithm $\mathcal{A}$ takes in X ($i.e. \mathcal{A}(X)$ as input and outputs a set of k centers $C = \{c_1, c_2, ..., c_k\}$. Consider the user, which will act as the adversary for the problem. The user only has knowledge of 1. the size, dimension, and other bounds of the input data, 2. the number of clusters k, and 3. the cluster labels for each example. The user has no knowledge of the values of the set C and X. The user's goal is to learn which regions of the input space are assigned to each cluster. Consider an sample of the original dataset $S = \{z_1, ..., z_m\} \subset X$ of size m and the maximum number of time-steps T. The question we ask is:

Question 3.1. Can the user (adversary) learn the bounds of the k clusters from the set C outputted from the algorithm $\mathcal{A}$ on the dataset X, from the addition of S points that we are allowed to query over T time-steps?

We demonstrate an adversarial procedure that provides an empirical solution to this problem. Our approach is as follows: we recast the data inference attack problem of learning k-centers as a multi-class classification problem under a noise label model, where the user is provided with at most $S \cdot T$ labeled examples and is asked to recreate the Voronoi diagram explaining the labeling. Rethinking the adversarial procedure as a classification provides the user the ability to construct decision boundaries around each of the k centers, as opposed to taking the mean of samples from a given label. Pseudocode for this approach is provided in Appendix A.

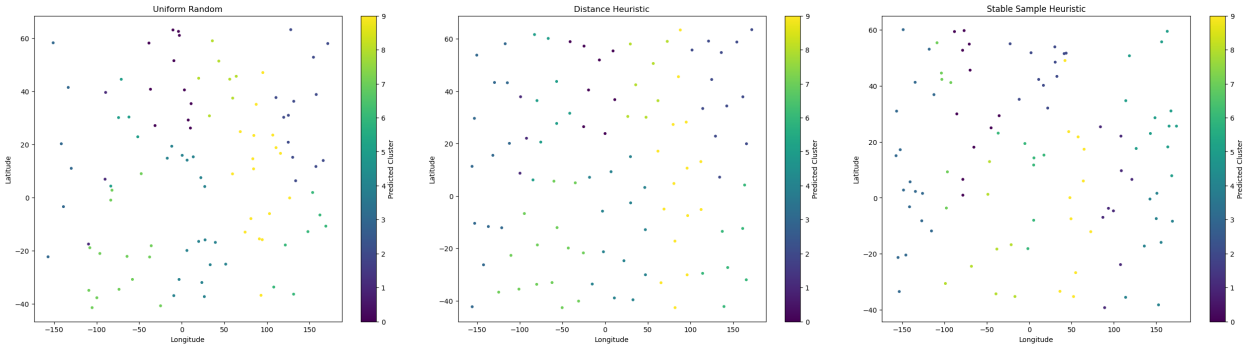
3.1 Sampling Heuristics

We experiment with different sampling heuristics for our inference attack algorithm, which are detailed as follows:

Uniform Sampling: Draw samples from a uniform distribution within the space $\mathbb{R}^d$.

Distance Sampling: Draw samples from a uniform distribution within the space $\mathbb{R}^d$ with the following condition: points are sampled from regions of space at least ϵ away from the query point of the previous iteration. This is done to prevent close points from being sampled consecutively as it can reduce clusters drifting toward those samples, thereby making boundary creation significantly harder.

Stable Sampling: Draw samples from a uniform distribution within the space $\mathbb{R}^d$ with the following conditions: if a sampled point has a k-means label which changes from one iteration of the algorithm to the next, we discard this point and choose a different point which is at least ϵ away from the previously sampled point which changed labels. This method builds on the distance sampling method but incorporates a routine to disregard points which change labels as these are close to decision boundaries and are less reliable for determining cluster centers. By doing this, the sampling technique ensures that sampled points are representative of the clusters as a point for which the label changes could be unreliable.



(a) Sampling with the *Random sampling* heuristic.

(b) Sampling with the *Distance-based sampling* heuristic.

(c) Sampling with the *Stable label-based sampling* heuristic.

Figure 2: Visualizing 200 samples produced using our two base sampling heuristics in the 2D space defined by Brightkite.

3.2 Query Models

One of the challenges in this attack is creating a labeled dataset that is large enough while reducing how much the cluster centers drift from their original positions. One also must budget the number of timesteps they are willing to take to create their training set. To explore this, we consider the following two query models in our experiments:

Oracle: In the Oracle Model, the sub-sampling from S procedure over T time steps will not affect the k centers in the set C in any way; that is, the algorithm A will not rerun during the sampling procedure. Since many of our methods focus on reducing the amount of noise in the labels, this query model serves as an upper bound for the noise reduction, as the clusters will never shift and therefore no noise is added.

Online: In the Online Model, for every time-step $t \in [T]$ and a subsample $S_t \subset S$ at that time-step, the algorithm $\mathcal{A}$ takes in as input $X' = X \cup S_t$ and outputs an updated set of k-centers C'. Essentially, the algorithm A reruns at each time step and with the addition of the new sub-sampled data points and the data points from the previous iteration. Over time, the clusters gradually shift from the original and target clusters. In our experiments, we use the online query model with the random and distance sampling heuristic.

4 Sample Complexity Lower Bounds

Previous results in the sampling complexity literature have provided lower and upper bounds for multi-class learning in the classical PAC setting. For our case, we will state and use the lower bound in [10], which is controlled by the Natarajan dimension, a combinatorial measure that generalizes the VC dimension [23] for the multi-class case. Expanding on the definitions for multi-class classification, let us define the agnostic sample complexity of a learning algorithm A is the function $m^a_{A,\mathcal{H}}$ defined as follows: for every $\epsilon, \delta > 0$, $m^a_{A,\mathcal{H}}(\epsilon, \delta)$ is the minimal integer such that for every $m \geq m^a_{A,\mathcal{H}}(\epsilon, \delta)$, every distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$,

$$Pr_{S_n \sim \mathcal{D}^n}[Err_{\mathcal{D}}(A(S_n)) > Err_{\mathcal{D}}(\mathcal{H}) + \epsilon] \leq \delta \tag{1}$$

We define $m^a_A(\epsilon, \delta) = \infty$ if no integer satisfies the previous inequality. $\mathcal{H}$ is learnable with A if for all ϵ and δ, the agnostic sample complexity is finite. The agnostic sample complexity of a class $\mathcal{H}$ is $m^a_{\text{PAC},\mathcal{H}}(\epsilon, \delta) = \inf_A m^a_{A,\mathcal{H}}(\epsilon, \delta)$ where the infimum is taken over all learning algorithms over $\mathcal{H}$. We say that a distribution $\mathcal{D}$ is realizable by a hypothesis class $\mathcal{H}$ if there exists some $h \in \mathcal{H}$ such that $Err_{\mathcal{D}}(h) = 0$. The realizable sample complexity of an algorithm A for a class $\mathcal{H}$, denoted as $m^r_{A,\mathcal{H}}$, is the minimal integer such that for every $m \geq m^r_{A,\mathcal{H}}(\epsilon, \delta)$ and every distribution $\mathcal{D}$ on $X \times Y$ which is realizable by $\mathcal{H}$, Equation (1) holds. The realizable sample complexity of a class $\mathcal{H}$ is $m^r_{\text{PAC},\mathcal{H}}(\epsilon, \delta) = \inf_A m^r_{A,\mathcal{H}}(\epsilon, \delta)$ where the infimum is taken over all learning algorithms over $\mathcal{H}$. Now, we can use the uniform sample complexity bounds of multi-class learning. Consider the shattering of a set S defined as the follows: for any partition of S into T and $S \setminus T$, there exists a $g \in \mathcal{H}$ whose behavior on T differs from its behavior on $S \setminus T$. The Natarajan dimension is as follows:

Definition 4.1. (Natarajan dimension) Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a hypothesis class and let $S \subseteq \mathcal{X}$. We say that $\mathcal{X}$ N-shatters S if there exists $f_1, f_2 : S \to \mathcal{Y}$ such that $\forall y \in S$, $f_1(y) \neq f_2(y)$, and for every $T \subseteq S$ there is a $g \in \mathcal{H}$ such that

$$\forall x \in T, g(x) = f_1(x), \text{and } \forall x \in S \setminus T, g(x) = f_2(x)$$

The Natarajan dimension of $\mathcal{H}$, denoted $d_N(\mathcal{H})$ is the maximal cardinality of a set that is N-shattering by $\mathcal{H}$.

This dimension coincides with the VC-dimension for $|\mathcal{Y}| = 2$. [10] then uses the Natarajan dimension to show a lower bound on the learner sample complexity:

Theorem 4.2. ([10]) For constants $C > 0$, for every hypothesis class $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ we have,

$$C\Big(\frac{d_N(\mathcal{H}) + \ln(\frac{1}{\delta})}{\epsilon}\Big) \leq m^a_{\text{PAC},\mathcal{H}}(\epsilon, \delta) \leq m^a_{A,\mathcal{H}}(\epsilon, \delta)$$

With this, consider $|\mathcal{Y}| = k$ to denote the k-center classes and the sample set $S_m \subset X$. Let us denote our inference attack learner as A_I. If A_I satisfies the definition of a learning algorithm, then for every $\epsilon, \delta > 0$, let $m = m^a_{A_I,\mathcal{H}}(\epsilon, \delta)$ be the agnostic sample complexity of a class of the learning algorithm A_i. Then for $C > 0$ sample complexity lower bound for our inference attack learner is

$$C\Big(\frac{d_N(\mathcal{H}) + \ln(\frac{1}{\delta})}{\epsilon}\Big) \leq m^r_{A_I,\mathcal{H}}(\epsilon, \delta)$$

However, these sampling complexity bounds for our data inference attack could be improved by considering the Rademacher Complexity instead of the Natarajan dimension. The reasoning is as follows: the Rademacher Complexity is distribution dependent and defined for any class real-valued functions, while the Natarajan and subsequently the VC-dimension are bounds that hold for any data distribution [26], [3], [4].

We note that this theoretical analysis is only relevant to our *oracle* querying strategy. In the online setting, the means produced by k-means will gradually drift from their initial setting, meaning that the labels for our queries will eventually differ from the labels according to the original clusters we are attempting to estimate. In future work, this analysis could be adapted to this setting by considering a noisy label model, where the amount of noise gradually increases as additional labels are drawn. This would almost assuredly increase the lower bound, as the presence of noise would make the initial Voronoi diagram more difficult to discern.

5 Data and Empirical Setup

5.1 Datasets

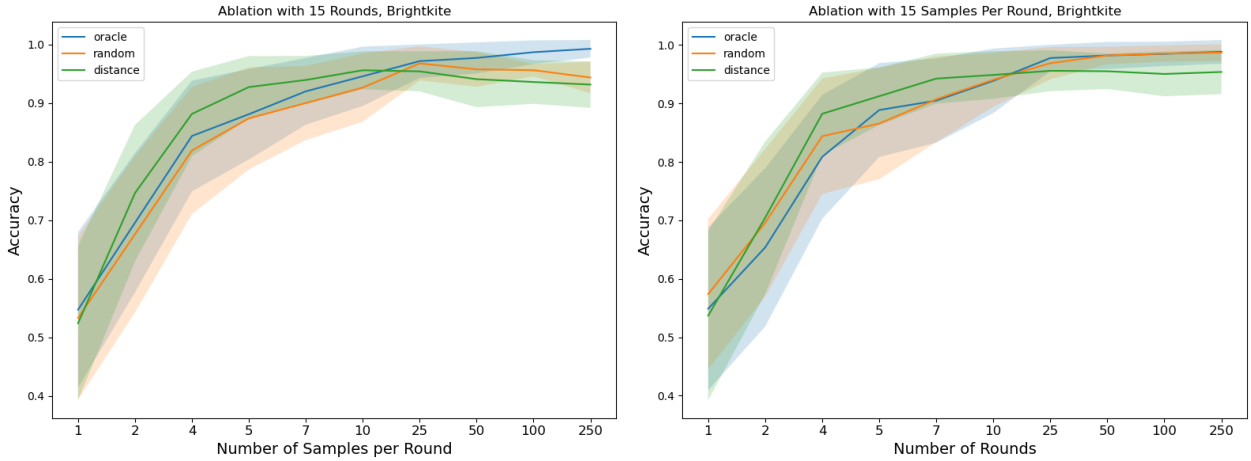
We experiment with three datasets for our model: Brightkite [15], Gowalla [15], and Rangequeries [11]. Brightkite and Gowalla are location datasets from the Bright-kite and Gowalla social networking service websites, where users shared their locations by "checking-in". Both Brightkite and Gowalla are two-dimensional datasets of latitude and longitude coordinates, where they each contain 4,491,143 check-ins and 6,442,890 check-ins of their users, respectively. Rangequeries is a dataset containing a set of range query workloads from Gaussian distributions over a real dataset on crime data from the city of Chicago. Rangequeries is an 8 dimensional dataset and contains 260000 examples.

For our experiments, our sample consists of 500 examples from each of these datasets to give query data a greater impact on k-means.

5.2 Experimental Setup

Each dataset was clustered using k-means into a specified number of clusters k. After convergence, the cluster assignment labels for data points in the training set were fixed. The adversarial setting assumes that the attacker only obtains cluster assignments for queried points, not their coordinates or any direct data attributes. Our objective is to approximate the original clustering structure, focusing on recovering the Voronoi partitions that the k-means model implicitly defines.

After training a base k-means clustering model as our target for attack, we perform 100 trials of conducting our attack with various sampling heuristics, and take the average of the accuracy on the unseen base dataset across all trials. Logistic regression models are trained on the extracted dataset using the Scikit-Learn python library [19].



(a) Experimenting with the number of samples taken per round.

(b) Experimenting with the number of total sampling rounds.

Figure 3: Varying the sampling procedure for the attack. We observe key differences in the behavior for large sample sizes for the oracle and our heuristics. Furthermore, the distance-based heuristic performs best in a low-sample regime.

6 Experimental Results

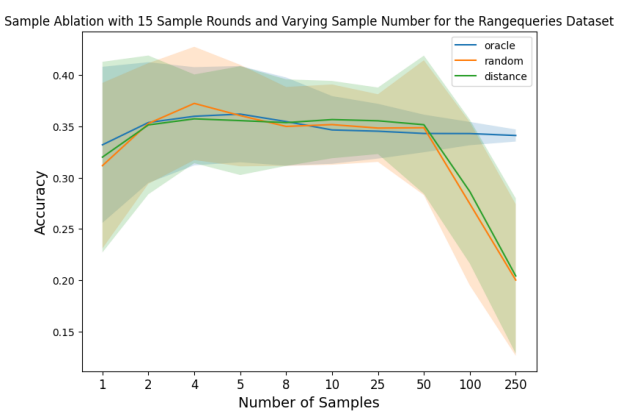
In this section, we detail the experimental setup and results that demonstrate the efficacy and limitations of our proposed data-inference attack on obfuscated k-means clustering outputs. We present the main findings regarding how sample size, sampling rounds, and the number of clusters influence the ability of our attack to recover cluster information. All experiments were designed to emulate practical scenarios where an adversary is restricted to observing only cluster assignments from a k-means trained model - often with added noise - to infer the underlying cluster structure.

6.1 Influence of Sample Size and Sampling Rounds

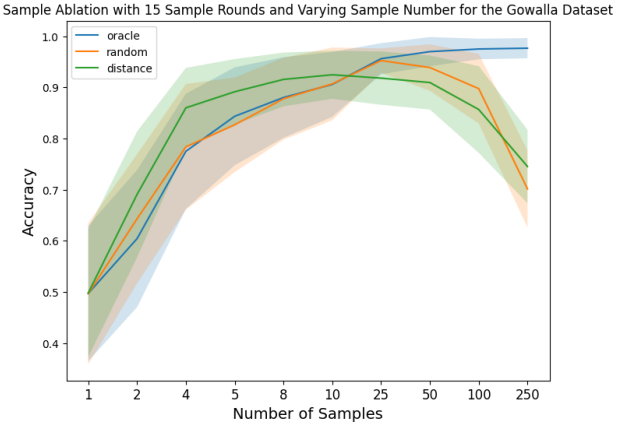
One set of experiments focused on how sample size and the number of sampling rounds influence the success of the inference attack. In these experiments, the adversary queries the clustering model in an iterative manner: each round involves querying a batch of points and then using the new information to refine the inferred cluster boundaries before proceeding to the next round.

The results (Figures 3, 4) demonstrate that increasing the total number of queried samples improves the reconstruction accuracy but with diminishing returns. After a certain threshold, adding more samples yields only marginal gains in accuracy. This plateau suggests that once major cluster boundaries are detected, additional queries primarily provide incremental refinements rather than substantial improvements.

Moreover, distributing queries over multiple rounds is generally more effective than gathering them all at once. Iterative refinement allows the adversary to identify gaps in their current understanding and strategically sample regions that remain uncertain. However, too many rounds with too few samples per round can prove inefficient. The data indicate that a balanced approach - a moderate number of rounds combined with an adaptive sampling strategy - strikes the best trade-off between effort and accuracy.



(a) Experimenting with the number of samples taken per round for Rangequeries.



(b) Experimenting with the number of total sampling rounds for Gowalla.

Figure 4: Varying the sampling procedure for the attack for two additional datasets. In these datasets, the detriment to performance caused by large sample sizes is much greater than that seen in Brightkite.

6.2 Comparison of Sampling Strategies

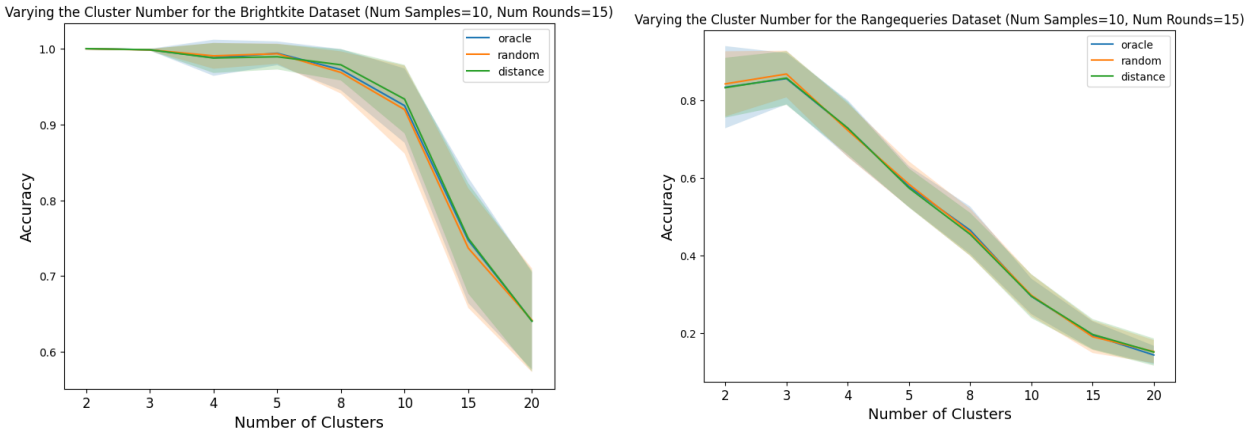
By overlaying their respective performance curves on the same plots, it becomes evident that distance-based sampling outperforms random sampling when sample budgets are limited. Because distance-based sampling avoids redundant queries in the same neighborhood, it more effectively explores the space, helping the adversary detect subtle shifts in cluster assignments.

Empirically, the distance-based approach attains higher accuracy with fewer queries, thus reaching a performance plateau more quickly than random sampling. The confidence intervals depicted in the plots suggest that distance-based sampling is also more consistent across runs, yielding fewer outlier cases with substantially worse performance. We note that distance-based clustering does not continue to improve in performance as the sample size grows as once a packing for the space is created, no further samples are extracted. This procedure could be changed in the future to allow for more samples to be extracted.

6.3 Influence of the Number of Clusters and Dimension

Another aspect of the problem examined in our experiments is how the number of clusters k affects the difficulty of the inference task. Intuitively, as k increases, the partitioning of the space grows more complex, and the adversary must discern a greater number of boundaries. Our results illustrate this trend (Figure 5): the inference attack becomes increasingly challenging as k grows. With the same number of queries, distinguishing between a larger set of smaller clusters is inherently more difficult than discriminating a few well-separated ones.

Our results also indicate that higher-dimensional datasets or those with heterogeneous feature scales pose additional challenges. Since the complexity of the cluster structure grows as the dimension of the dataset is higher, the density of samples must be much greater to approximate cluster boundaries reliably. This can be inferred in the decay of performance between the Rangequeries dataset than for the Brightkite dataset, since Rangequeries is an 8 dimensional dataset compared to the 2-dimensional dataset. However, the strong performance of our attacks at two or three



(a) Changing the number of clusters for the Brightkite dataset.

(b) Changing the number of clusters for the Rangequeries dataset.

Figure 5: Accuracy of the attack on two separate datasets as the number of clusters increases. The degree to which the performance decays, as well as the rate at which the decay occurs, varies greatly between datasets.

clusters also implies that the data generative procedure behind Rangequeries may make our query strategies ineffective when there is a high number of clusters.

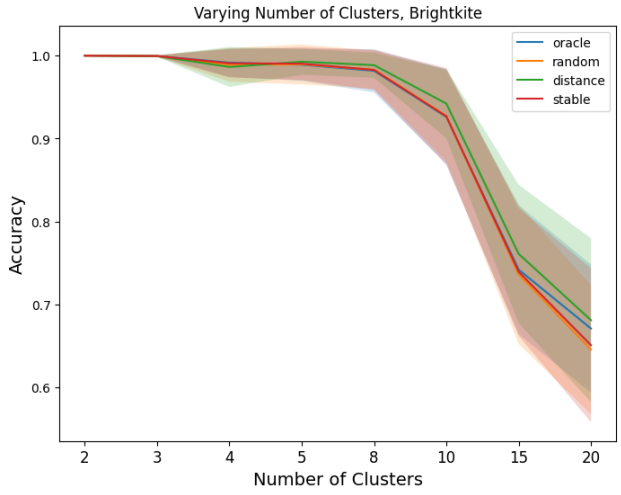
Our results also indicate that higher-dimensional datasets or those with heterogeneous feature scales pose additional challenges. Since the complexity of the cluster structure grows as the dimension of the dataset is higher, the density of samples must be much greater to approximate cluster boundaries reliably. Similarly, if certain features dominate due to scaling differences, clusters may be elongated or skewed, making it harder to infer partitions from sparse samples.

Our experiments show that while performance degrades under these conditions, the relative benefit of adaptive sampling approaches persists. Although gains may be more modest, they remain significant compared to naive random strategies.

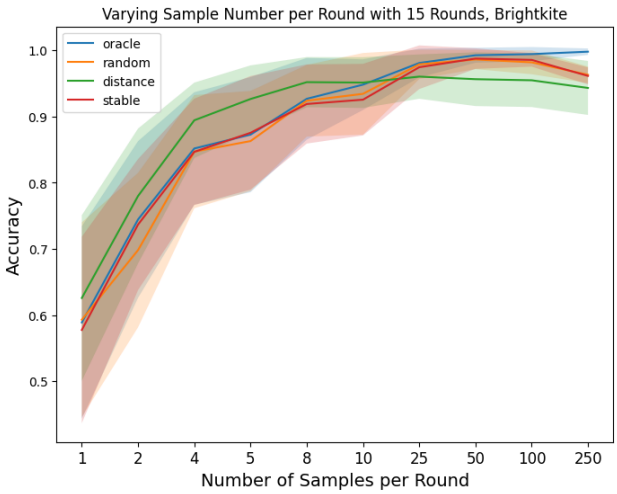
6.4 Improving the Heuristic and Introducing Noise

As the number of clusters increases, both the stable-label-based sampling and distance-based sampling methods exhibit a decline in accuracy. Under low-cluster conditions, both methods perform similarly. As the cluster count grows, however, the decision boundaries between clusters become more numerous and potentially closer to each other. In these conditions, the stable method often maintains a more consistently high accuracy than the distance-based approach, or at least degrades more gradually. When examining the influence of sample size, we observe that increasing the number of samples per round leads to significant performance improvements for all strategies—oracle, random, distance, and stable. The stable and distance methods begin relatively close together, but as the sample size grows, stable sampling tends to improve at a steady pace. The stable method's advantage becomes more apparent when enough samples are gathered: it not only secures a high level of accuracy but also reduces the uncertainty reflected by the shaded confidence intervals.

The introduction of Gaussian noise into the underlying data affects these dynamics by increasing the ambiguity of cluster boundaries and effectively making the learning task harder. Under noisier conditions, all methods experience some degree of accuracy reduction as the true cluster structure

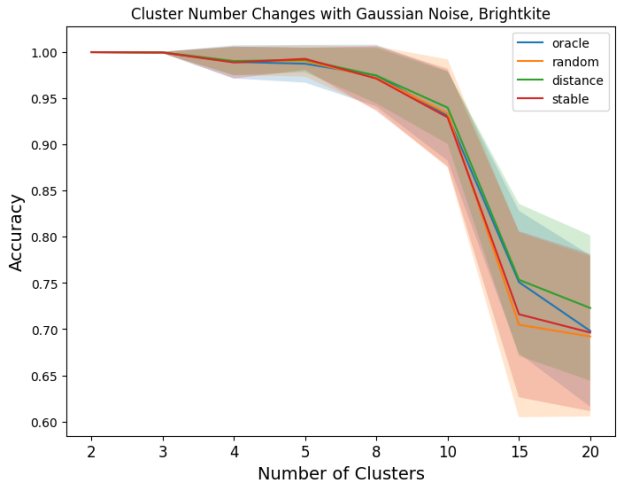


(a) Changing the number of clusters for the Brightkite dataset.

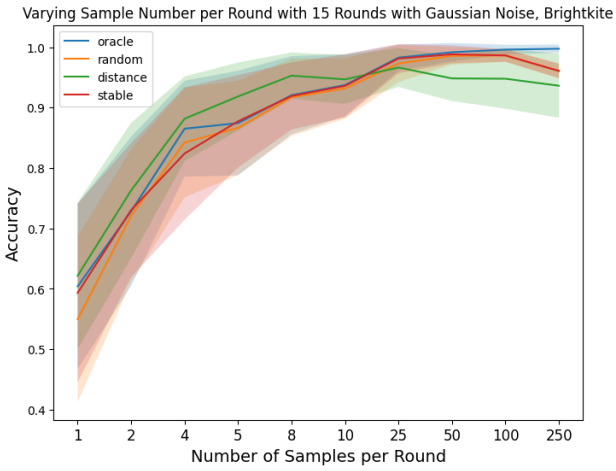


(b) Changing the number of samples for Brightkite

Figure 6: Accuracy of the attack with varying numbers of samples and clusters.



(a) Changing the number of clusters for Brightkite



(b) Changing the number of samples for Brightkite

Figure 7: Accuracy of the attack with varying numbers of samples and clusters.

becomes less distinct. However, the contrast between stable and distance approaches becomes more pronounced. Although both curves descend in accuracy with growing numbers of clusters or fewer samples, the stable approach generally retains a more favorable accuracy level or at least remains more resilient as noise levels increase.

6.5 Discussion of Results

The experimental results lead to several key insights:

1. **Iterative, informed sampling outperforms naive methods:** Using feedback from previous rounds of queries enables the adversary to more efficiently locate cluster boundaries. Distance-based heuristics, in particular, exceed oracle performance in certain regimes.

2. **Trade-offs between sample size and number of rounds:** While larger sample sets and iterative queries both help improve accuracy, each has diminishing returns. An intermediate approach with several moderate rounds and intelligent sampling provides a good balance.

3. **Challenges of increasing complexity:** As the number of clusters k grows, or as dimensionality and feature heterogeneity increase, the inference task becomes harder. Although performance declines, adaptive strategies still outperform random sampling.

4. **Generalizability and practical implications:** The results confirm that even revealing only cluster assignments (as opposed to raw data points) can leak significant information. This highlights the need for privacy-preserving clustering methods and emphasizes that adaptive querying strategies must be considered in threat models for secure data release.

In summary, the experiments highlight both the potential and the limitations of data-inference attacks against k-means clustering outputs. While certain conditions make inference more challenging, thoughtful query strategies consistently yield better approximations of the underlying cluster structure than naive approaches. These findings provide a foundation for future work aimed at developing both stronger inference attacks and robust defense mechanisms in privacy-sensitive clustering applications.

7 Conclusion

This work explored the possibility of extracting the associated partitions of input space for a k-means clustering under an extremely limited threat model. We find that even in the online setting where the cluster set C was shifting over consecutive time-steps, our learning algorithm attack was able to infer the implied Voronoi diagram with accuracy comparable to an Oracle method for each dataset we ran it on. Changes to the sampling method can also improve performance by up to 5 % percent in certain regimes. This improvement, as well as the demonstration that introducing too many query samples in a single sampling round can produce noisy labels (Figure 3), provides a starting point for future work. Furthermore, we find issues with the efficacy of this attack in higher dimensions, especially when the number of clusters to infer is high (Figure 5). Future work should theoretically and empirically investigate how both the dimensionality of the space, scale of input features, and data generative procedure interplay to determine the performance of this attack method.

For the empirical side, future work should look into restricted means of sampling, changing the underlying training data X, and studying different clustering settings, such as k-median, k-centers, Mixtures of Gaussians, spectral clustering, and more. For the theoretical side, future work should look into tighter bounds for the sampling complexity of our data inference attack algorithm, whether using the Natarajan dimension (which is distributional independent of the underlying data) or the Rademacher complexity (which is distributional dependent of the underlying data).

In conclusion, our work sheds light on how a seemingly harmless practice that several companies engage in may actually pose a threat to their users' privacy. We are hopeful that by demonstrating this, future approaches may be developed to help stymie attacks such as the one presented in this work.

References

[1] N. Ailon, R. Jaiswal, and C. Monteleoni. Streaming k-means approximation. *Advances in neural information processing systems*, 22, 2009.

[2] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[3] M. Balcan and C. Berlind. Cs 8803- machine learning theory: Rademacher complexity lecture notes. *https://www.cs.cmu.edu/ ninamf/ML11/lect1117.pdf*, 2011.

[4] P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[5] R. Bhattacharjee, J. Imola, M. Moshkovitz, and S. Dasgupta. Online k-means clustering on arbitrary data streams. In *International Conference on Algorithmic Learning Theory*, pages 204–236. PMLR, 2023.

[6] Z. Bodó, Z. Minier, and L. Csató. Active learning with clustering. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, pages 127–139. JMLR Workshop and Conference Proceedings, 2011.

[7] V. Cohen-Addad, B. Guedj, V. Kanade, and G. Rom. Online k-means clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 1126–1134. PMLR, 2021.

[8] V. Cohen-Addad, H. Esfandiari, V. Mirrokni, and S. Narayanan. Improved approximations for euclidean k-means and k-median, via nested quasi-independent sets. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1621–1628, 2022.

[9] E. Creager and R. Zemel. Online algorithmic recourse by collective action. *arXiv preprint arXiv:2401.00055*, 2023.

[10] A. Daniely, S. Sabato, S. Ben-David, and S. Shalev-Shwartz. Multiclass learnability and the erm principle. *J. Mach. Learn. Res.*, 16(1):2377–2404, 2015.

[11] D. D. Dheeru and E. K. Taniskidou. Uci machine learning repository, 2017.

[12] A. Epasto, V. Mirrokni, S. Narayanan, and P. Zhong. k-means clustering with distance-based privacy. *Advances in Neural Information Processing Systems*, 36, 2024.

[13] T. Finley and T. Joachims. Supervised k-means clustering, 2008.

[14] Y. Huang, Z. Huo, and Y. Fan. Dra: A data reconstruction attack on vertical federated k-means clustering. *Expert Systems with Applications*, 250:123807, 2024.

[15] L. Jure. Snap datasets: Stanford large network dataset collection. *Retrieved December 2021 from http://snap. stanford. edu/data*, 2014.

[16] E. Liberty, R. Sriharsha, and M. Sviridenko. An algorithm for online k-means clustering. In *2016 Proceedings of the eighteenth workshop on algorithm engineering and experiments (ALENEX)*, pages 81–89. SIAM, 2016.

[17] N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM journal on computing*, 13(1):182–196, 1984.

[18] H. T. Nguyen and A. Smeulders. Active learning using pre-clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 79, 2004.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[20] J. Perdomo, T. Zrnic, C. Mendler-Dünner, and M. Hardt. Performative prediction. In *International Conference on Machine Learning*, pages 7599–7609. PMLR, 2020.

[21] J. Ren, J. Xiong, Z. Yao, R. Ma, and M. Lin. Dplk-means: A novel differential privacy k-means mechanism. In *2017 IEEE Second International Conference on Data Science in Cyberspace (DSC)*, pages 133–139. IEEE, 2017.

[22] O. Reyes, C. Morell, and S. Ventura. Effective active learning strategy for multi-label learning. *Neurocomputing*, 273:494–508, 2018.

[23] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity: festschrift for alexey chervonenkis*, pages 11–30. Springer, 2015.

[24] C. Xia, J. Hua, W. Tong, and S. Zhong. Distributed k-means clustering guaranteeing local differential privacy. *Computers & Security*, 90:101699, 2020.

[25] S. Xiong, J. Azimi, and X. Z. Fern. Active learning of constraints for semi-supervised clustering. *IEEE Transactions on Knowledge and Data Engineering*, 26(1):43–54, 2013.

[26] D. Yin, R. Kannan, and P. Bartlett. Rademacher complexity for adversarially robust generalization. In *International conference on machine learning*, pages 7085–7094. PMLR, 2019.

[27] X. Zhang, X. Zhu, and L. Lessard. Online data poisoning attacks. In *Learning for Dynamics and Control*, pages 201–210. PMLR, 2020.

A Algorithm Pseudocode

Algorithm 1 Attack Experiment

Require: Data D, initial KMeans model K, rounds R, samples per round S, method M, retraining R_m, repeats N, ϵ, clusters C, noise flag G

Ensure: Accuracy list *acc_list*

bounds ← determine_bounds(D)
acc_list ← []
for *repeat* ∈ $\{1, \ldots, N\}$ **do**
  new_data ← [], *new_labels* ← []
  new_kmeans ← copy(K)
  prev_kmeans ← copy(*new_kmeans*)
  for *iteration* ∈ $\{1, \ldots, R\}$ **do**
    if G **then**
      noise ← GaussianNoise($0, 0.1,$ shape(D))
      $data\_for\_round \leftarrow D + noise$
    else
      $data\_for\_round \leftarrow D$
    end if
    old_samps ← (concatenate(*new_data*) if not empty, else *None*)
    if $M =$ 'random' **then**
      if $R_m =$ 'online' **then**
        $(re\_new\_samps, re\_new\_labels, new\_kmeans) \leftarrow$ gen_new_labels($data\_for\_round, S, bounds,$ M, R_m, new_kmeans, C)
      else
        $(re\_new\_samps, re\_new\_labels) \leftarrow$ gen_new_labels($data\_for\_round, S, bounds, M, R_m,$ new_kmeans, C)
      end if
    else if $M =$ 'distance' or $M =$ 'stable_random' **then**
      if $R_m =$ 'online' **then**
        $(re\_new\_samps, re\_new\_labels, new\_kmeans) \leftarrow$ gen_new_labels($data\_for\_round, S, bounds,$ M, $R_m, new\_kmeans, old\_samps, \epsilon, C, prev\_kmeans$)
      else
        $(re\_new\_samps, re\_new\_labels) \leftarrow$ gen_new_labels($data\_for\_round, S, bounds, M, R_m,$ new_kmeans, old_samps, ϵ, C)
      end if
    else
      error: invalid method
    end if
    Append $re\_new\_samps, re\_new\_labels$ to new_data, new_labels
    $prev\_kmeans \leftarrow$ copy(new_kmeans)
  end for
  $new\_data \leftarrow$ concatenate(new_data)
  $new\_labels \leftarrow$ concatenate(new_labels)
  $classifier \leftarrow$ train_log_reg(new_data, new_labels)
  $acc \leftarrow$ accuracy_score($K.labels\_, classifier.predict(data\_for\_round)$)
  Append *acc* to *acc_list*
end for
return *acc_list*