# Implementation and Improvement of a Non-parametric Information Theoretic Clustering Algorithm

Fan Liang
*Georgetown University*
*Data Science and Analytics Department*
Washington, D.C.
fl430@georgetown.edu

Xiner Ning
*Georgetown University*
*Data Science and Analytics Department*
Washington, D.C.
xn11@georgetown.edu

***Abstract*—In this paper we implement the nonparametric information theoretic clustering algorithm proposed in Faivishevsky and Goldberger's paper. We apply the algorithm in both generated and real-world data with various dimensions and structures. Algorithm performance is further compared with that of K-means. After analyzing drawbacks of the algorithm, we propose alternatives of estimating cluster entropy and reducing computational complexity to improve the performance.**

## I. Introduction

Clustering is a powerful and commonly-used exploratory data analysis technique on unsupervised data. Clustering algorithms reveal structure of the data by identifying homogeneous subgroups with similar data points. As a popular technique in machine learning, clustering is used in a variety of applications such as detecting fake news, market segmentation, document analysis and image compression.

To make effective and automatic grouping of objects, we need to tackle two main technical issues. Firstly, the clustering algorithm should be capable of producing both linearly and non-linearly separated clusters. Classic clustering algorithm such as K-means does a poor job when handling complex high-dimensional real-world data with complicated geometric shapes. A robust clustering algorithm should group data points to the same cluster even though they are far away from each other. Secondly, the clustering algorithm should not rely on any explicit parametric representation of data points. In general cases, we won't have any prior knowledge in neither distribution of individual data points nor any parametric model of the within-cluster distribution. So an effective clustering algorithm should make no assumption in feature distribution of objects. In this paper, we will talk about clustering algorithms that overcome these two obstacles.

## II. Related Work

In Faivishevsky & Goldberger's paper, they propose a novel clustering algorithm based on maximizing the mutual information between data points and clusters. Instead of assuming the data are given in terms of distributions or impose any parametric model in the with-in cluster distribution, they estimate the average cluster entropy and find a clustering method that maximizes the mutual information between data points and clusters. In this section, we will explain in details the proposed Nonparametric Information Theoretic Clustering algorithm by Falvishesky and his colleagues. In section 4 and 5, we will describe how we implement the model in simulated data and compare model performance with that of K-means.

Before introducing NIC, the paper discusses mutual information criterion of clustering in detail.The task of clustering a set of data points X is to find a clustering method $C(X)$ that optimizes clustering score function $S(C)$ (Faivishevsky  Goldberger 2010). Since $I(X;C) = H(X) - H(X|C)$ and $H(X)$ does not depend on the specific clustering, they use mutual information score function

$$S_{MI}(C) = H(X|C) = \sum_{j=1}^{n_c} \frac{n_j}{n} H(X|C=j)$$

to measure cluster entropy.

In order to benefit from the mutual information clustering score function, the paper uses a nonparametric estimation of in-cluster entropy without having an explicit representation of the intra-cluster distribution. Traditional methods used to estimate mutual information are either restricted to low-dimensional dataset or lead to a certain lack of smoothness of estimator value as a function of the sample coordinates. To overcome the drawbacks, the paper utilizes the MeanNN differential entropy by averaging all k estimators from k-nearest neighbor estimations. MeanNN estimator computes the pairwise distance between all the given data points. With this estimator, the score function now becomes

$$S_{NIC} = \sum_i \frac{d}{n_j - 1} \sum_{i \neq l | c_i = c_l = j} \log \parallel x_i - x_l \parallel$$

The paper optimizes the score function by applying a greedy sequential algorithm that is similar to the k-means algorithm. It starts with a random partition of data points and for each data point, it checks whether moving it from its current cluster to another one improves the score function $S_{NIC}(C)$. The loop continues until reaching a local optimum. So depending on the initial partition, this algorithm returns different results. Computational complexity is $O(n^2)$ since it needs to calculate the log-distance of data point $x_i$ to all the other points in the cluster.

## III. Datasets

For this project, we generated our own data in different shapes and clusters to train the clustering model. First we drew random samples from a multivariate normal distribution. Given a random variable $z$ with one distribution, we can create another random variable $X = g(z)$ with a completely different distribution (Doersch 2016). We then used the function $g(z) = z/10 + z/||z||$ suggested in Doersch's paper to form a Gaussian Ring.

## IV. Methods and Algorithms

In this project, we aim to implement Faivishevsky and Goldberger's nonparametric information clustering algorithm. Given limited code sources on this topic, we wrote our own Python script.

The basic logic of the NIC algorithm is:

- Whiten data by covariance matrix.
- Randomly assign all points to a cluster.
- Calculate the $S_{NIC}$ score using the entropy estimator based on pair-wise distance. We used euclidean distance in this project:

$$S_{NIC} = \sum_i \frac{1}{n_j - 1} \sum_{i \neq l | c_i = c_l = j} \log \parallel x_i - x_l \parallel^2$$

- In a circular manner, go over all points by reassigning the point to different clusters and update the current assignment by choosing label that leads to the minimal score.
- Repeat step 3 until no point change to another cluster or reach the pre-set iteration number.

In the testing part, we found if we implemented the data whitening by multiplying the generated data by the matrix $Cov(X)^{-\frac{1}{2}}$, we got imaginary numbers in the matrix. So we decided to use pure covariance matrix instead and that worked out well. Similar to what the paper had claimed, we also proved that clustering results were affected by different choices of initial points. In order to get the smallest cluster score, we implemented a multiprocessing API in our fit function so we could minimize the time of testing with different initial points.

## V. Results

We designed a NIC class in Python and successfully implemented the NIC algorithm proposed in Faivishevsky and Goldberger's paper. In this section, we will compare model performances for both NIC and K-means using generated data as well as a real-world dataset.

First of all, we generated 3 clusters of data points following Gaussian distribution to compare NIC algorithm with K-mean clustering algorithm. We used scikit-learn API to implement K-mean clustering. As shown in Fig.1 above, we see these two algorithms give the same results. However, NIC clustering algorithm is running much slower than K-mean. Because our package is not as well-optimized as the sikit-learn package. We will discuss more details about this topic in next section.

Then we generated data with a more complicated structure, as shown in Fig.3 below. We brought in Gaussian Ring and mixed it with some basic Gaussian distribution plots. We see that K-mean clustering algorithm fails when each cluster has different

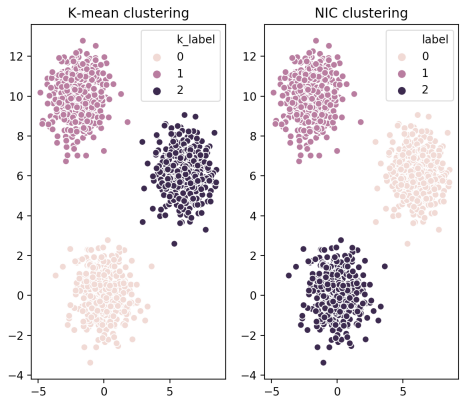


Fig. 1. Example of performance of K-mean clustering and NIC clustering on general case.

density. K-mean fails to identify complex structure in data because it clusters data points purely on their geometric closeness to the cluster centroid (Raykov, Boukouvalas, Baig, Little 2016). On the contray, NIC clustering ignores shape and distribution of the data and successfully identify the structure.

We also applied the algorithm on the Wine dataset from UCI ML Repository. Performances were evaluated using the Rand Index score (Rand 1971). NIC outperformed K-means in classifying the wine dataset as Fig.2 shows below.

## VI. Analysis, Discussion and Future Work

In this section, we will discuss some disadvantages of the NIC algorithm.

Firstly, NIC has a low running speed. To show how much time it takes to run the model, we applied the algorithm on datasets in various dimension and numbers of clusters. For each cluster, we generated 500 data points. Furthermore, we compared the running time with that of K-menas using the same dataset. Results are shown in TABLE 1 below. In the previous sections, we implemented the K-mean clustering algorithm using API from scikit-learn package. In order to get a fair result (most core algorithms in scikit-learn are written in Cython which is way faster than Python), we implemented a Python version K-mean algorithm, based on the source code included in Imad Dabbura's article K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks, to compare the

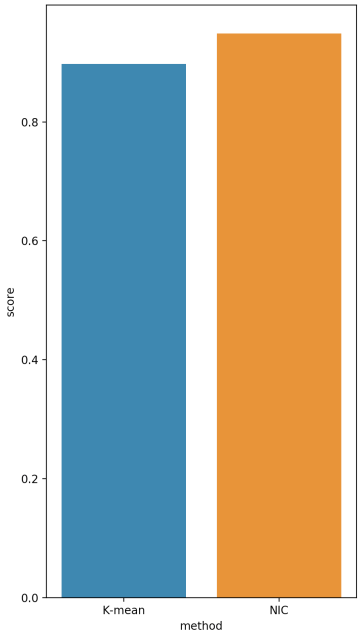


Fig. 2. Example of performance of K-means clustering and NIC clustering in Wine dataset from UCI Machine Learning Repository

running time (Dabbura 2019).

| Dim | Cluster | t(Kmean) | t(NIC) |
|---|---|---|---|
| 2 | 2 | 0.0006 | 7.42 |
| 2 | 3 | 0.0016 | 48.69 |
| 2 | 4 | 0.002 | 137.23 |
| 3 | 2 | 0.0006 | 30.57 |
| 3 | 3 | 0.0012 | 78.24 |
| 3 | 4 | 0.0019 | 196.42 |
| 4 | 2 | 0.006 | 10.51 |
| 4 | 3 | 0.009 | 118.52 |
| 4 | 4 | 0.007 | 563.97 |

TABLE I
Running time comparison.

We can see the NIC cluster algorithm (time complexity $O(n^2)$) is way more slower than the K-mean clustering algorithm (time complexity $O(nKId)$). We think this is because:

- Our algorithm is not well optimized.
- In each cluster reassigning, the calculation in NIC algorithm is much more complicated than that of the K-means.

Secondly, results of running NIC algorithm on the same data vary depending on the values chosen as

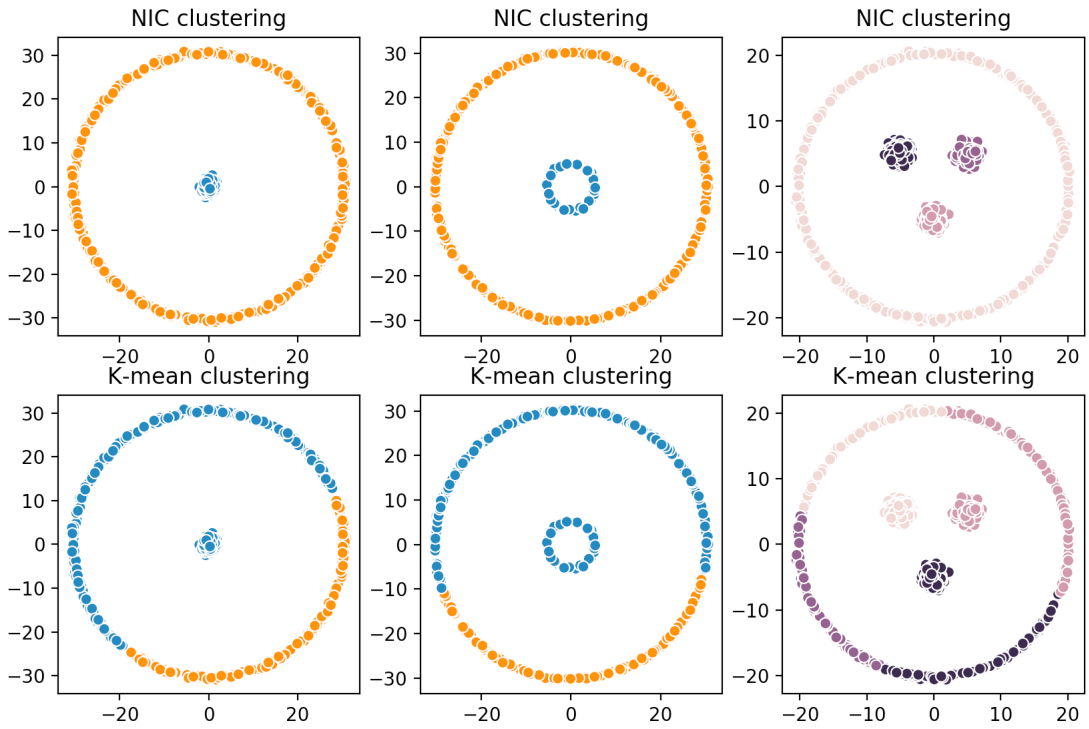


Fig. 3. Comparison of NIC clustering against K-mean clustering. (First row NIC, second row K-mean)

initial starting points. A bad initial point will prevent the algorithm from converging into its global minimum. As shown in Fig. 5, we train the NIC model with 9 different initial clustering labels using the same set of data. And we've find one false output already. Since the NIC algorithm can be significantly improved by repeating the algorithm, we cannot optimize its running time.

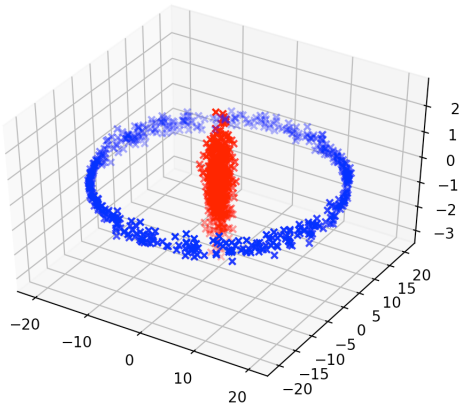

Fig. 4. Example of the performance of NIC algorithm in 3-D.

Thirdly, the NIC algorithm does not perform well in dataset with dimensions higher than 3. We applied the algorithm in a 3-D dataset (Fig.4). However, NIC does not give great results on data with more features.

## VII. Conclusion and Future Improvement

We implemented the NIC algorithm proposed by Faivishevsky and Goldberger in their paper in Python. This algorithm is better than K-means in identifying non-linear structure in data. As opposed to other clustering methods based on information theory, NIC does not require any prior knowledge of the within cluster distribution. However, NIC has a lower running speed and requires running repeatedly to reach the global minimum.

Based on the drawbacks we found about the NIC algorithm, we think some future works can be done to improve this clustering model:

- Change the entropy based estimator. The core of this algorithm is the entropy estimator based on pair-wise distance. There are a number of non-parametric techniques for the differential entropy estimation of random vectors $x_,...,x_n \in Rd$ which

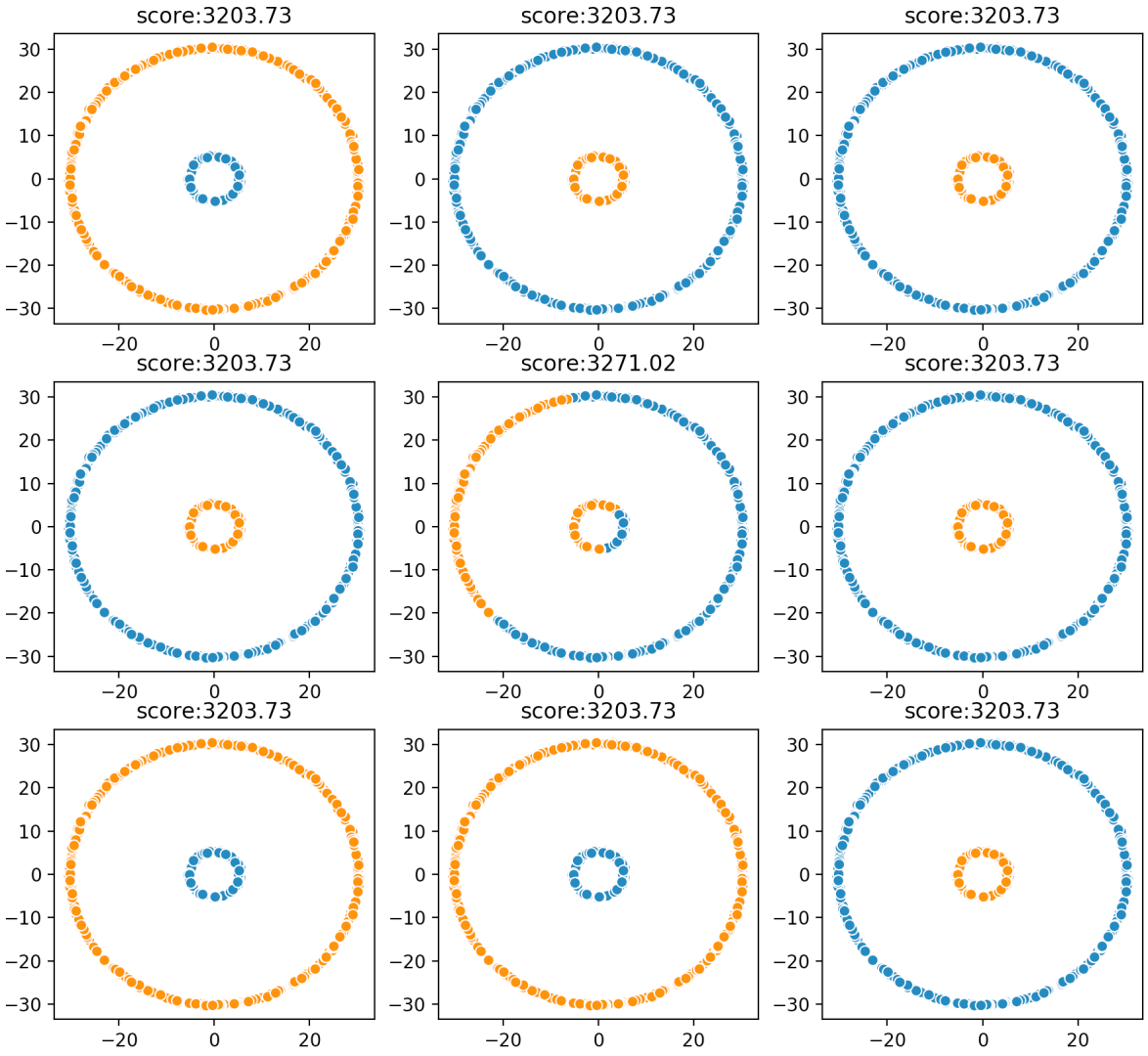


Fig. 5. Result with different initial cluster labels.

are all variants of $H_k = \frac{d}{n}\sum_{i=1}^{n} log\epsilon_i k + const(k)$ (Delattre Fournier 2017). We have tried to replace the estimator but we did not find a proper one that improves the algorithm.

- Use different data whitening method. Our proposed use of pure covariance matrix has improved the model performance. However, we do not claim that this is the best data whitening method to use.
- Reduce the algorithm running time by updating the cluster assignment before a single loop is finished. NIC uses a greedy sequential algorithm. This algorithm updates the cluster labels only after a single loop has finished. So we think an algorithm that updates the cluster labels more than once in a single loop will reduce the running time significantly.

## REFERENCES


[1] Doersch, Carl. "Tutorial on Variational Autoencoders." 16 Aug. 2016, https://arxiv.org/pdf/1606.05908.pdf.

[2] Delattre, Sylvain, and Nicolas Fournier. "On the Kozachenko–Leonenko Entropy Estimator." Journal of Statistical Planning and Inference, vol. 185, 2017, pp. 69–93., doi:10.1016/j.jspi.2017.01.004.

[3] Dabbura, Imad. "K-Means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks." Medium, Towards Data Science, 3 Sept. 2019, towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a.

[4] Faivishevsky, Lev, and Jacob Goldberger. "A Nonparametric Information Theoretic Clustering Algorithm." Aug. 2010.

[5] Raykov, Yordan P., et al. "What to Do When K-Means Clustering Fails: A Simple Yet Principled Alternative Algorithm." Plos One, vol. 11, no. 9, 2016, doi:10.1371/journal.pone.0162259.

[6] Rand, William M. "Objective Criteria for the Evaluation of Clustering Methods." Journal of the American Statistical Association, vol. 66, no. 336, Dec. 1971, pp. 846–850., https://www.jstor.org/stable/2284239.