Towards an Optimal Subspace for K-Means:
Implementation of algorithm applied on various datasets

Final Report

for

COMP5331 Knowledge Discovery in Database

Hyunjung G. Kwak Young D. Kwon Kirill A. Shatilov

Table of Contents

1. Introduction

Due to the curse of dimensionality, analyzing high-dimensional space data have been an issue in terms of clustering. The paper proposed an optimal dimensionality reduction of SubKmeans which is an improved algorithm from the original k-means algorithm, with revealing the prominent cluster structure hidden from subspace concept. With reaching to eigendecomposition, the algorithm get the minimal cost function which is the sum of expectations of errors in all clustered spaces and noise space.

For experiments, the authors used two preprocessing methods which are widely used such as Principal Component Analysis (PCA) and Independent Component Analysis (ICA) and four proposed algorithms such as LDA-k-means, FOSSCLU, ORCLUS and 4C after standardization mostly and compared the results with real-world datasets.

In this implementation project, all the concepts being used is discussed and algorithms are also implemented with several languages such as Python and Java. Kernel Trick is additionally studied to compare the performance of Sub-Kmeans. With Principal Component Analysis (PCA), Independent Component Analysis (ICA) and Linear Discriminant analysis (LDA), the data used in the paper, 2D data points and more big data sets are tested to see the importance of preprocess and the effect of the number of instances.

2. Related Work

It is common to pre-process multidimensional data using Principal Component Analysis (**PCA**)[1] before the application of a full-space clustering technique. It reveals the directions of most variance within this dataset. Over the topic of multidimensional clustering, it works as a state-of-the-art, being compared to most of the proposed algorithms.

Another preprocessing technique is **ICA** (Independent Component Analysis)[2], is not directly a dimensionality reduction mechanism, but aims at identifying statistically independent sources within the data.

Isomap[3] is method, which finds a nonlinear lower dimensional representation based on the estimated geometric properties of the data's manifold.

ISAAC[4] is a generalized subspace clustering technique capable of finding multiple nonredundant clusterings in arbitrarily-oriented subspaces. It uses Independent Subspace Analysis (ISA) to find the subspace collection that minimizes the statistical dependency (redundancy) between clusterings. ISAAC uses EM clustering, that makes algorithm user-parameter free. ISAAC is full-cycle clustering technique producing significant result on chosen sample datasets.

Kernel method[5] is an algorithm for high-dimensional, nonlinear implicit feature space data with using an inner product in a feature space. It is mostly well known for operating with support vector machines (SVMs), but it is also transcendent to learn a nonlinear decision boundary with principal components analysis (PCA), regressions and spectral clustering. This approach is called the kernel trick and with its poor performance on scaling to large numbers of training samples or features in the input space, several approximations to kernel methods were researched in this implementation project. We used Radial Basis Function (RBF) kernel, Additive Chi Squared (ACS) kernel and Skewed Chi Squared (SCS) kernel are used to explicitly model kernel maps.

X-means[6] is well known extension of Kmeans clustering algorithm. X-means proposes efficient and automated estimation of the number of clusters to tackle one of the major problems of Kmeans.

3. Method & Algorithm

3.1 Description of any mathematical background necessary for our problem

Mathematical background is not a must. But broad knowledge about Linear Algebra is required to understand derivation used in this paper.

Trace Property

In the linear algebra, the trace operator is invariant to the transpose operator.

$$Tr(A) = Tr(A^T)$$

The trace of a square matrix composed of many factors is also invariant to moving the last factor into the first position.

$$Tr(ABC) = Tr(CAB) = Tr(BCA)$$

Rigid Transformation

A rigid transformation (isometry) of a vector space is a transformation that preserves distances between every pair of points. A rigid transformation is formally defined as a transformation that, when acting on any vector v, produces a transformed vector $T(v)$ of the form:

$$Tr(v) = Rv + t$$

where transposed R equals to R inversed (i.e., R is an orthogonal transformation), and t is a vector giving the translation of the origin.

A proper rigid transformation has, in addition:

$$det(R) = 1$$

which means that R does not produce a reflection, and hence it represents a rotation (an orientation-preserving orthogonal transformation). Indeed, when an orthogonal transformation matrix produces a reflection, its determinant is –1.

Projection

A projection is a linear transformation P from a vector space to itself, such that applied twice to any value, it gives the same result as if it were applied once (idempotent).

Eigendecomposition

In linear algebra, eigendecomposition or sometimes spectral decomposition is the factorization of a matrix into a canonical form, whereby the matrix is represented in terms of its eigenvalues and eigenvectors. Only diagonalizable matrices can be factorized in this way.

Kernel methods

Kernel method is an algorithm for non-linearly classifiable data to become linearly separable, helping with a feature map $\{\varphi : \chi \rightarrow \nu\}$ which satisfies $k\{x, x'\} = <\varphi(x), \varphi(x')>_\nu$ and two of the popularly used one are polynomial kernel $k(\vec{x_i}, \vec{x_j}) = (\vec{x_i} \cdot \vec{x_j})^2$ and gaussian radial basis kernel $k(\vec{x_i}, \vec{x_j}) = \exp(-\gamma ||\vec{x_i} - \vec{x_j}||^2)$. Kernel-trick methods seek a certain dimension which helps data can be linearly separable but an approximations to kernel methods (ACS and SCS kernel) $<z(x), z(x')> \approx <\varphi(x), \varphi(x')> = k\{x, x'\}$ can even reach to explicitly model kernel maps and by far highly performed implicit feature space learning which is poor to scaling to large numbers of training samples or features in the input space.

3.2 Formal description of any important algorithms used

3.2.1 K-means

SubKMeans is built on top of original k-means. Given an initial set of *k* means, k-means algorithm proceeds by alternating between two steps:

Assignment step: Assign each data point to the cluster whose mean is intuitively the "nearest". It is common to use euclidian metric distance in order to estimate the distance between data point and cluster mean.

Update step: Calculate new mean for each of k clusters. In the original version of algorithm average coordinates of all data point belonging to cluster are considered to be new cluster mean. So far multiple improvements are proposed to method of calculating new mean, such as k-centroids.

The algorithm has converged when the assignments no longer change. There is no guarantee that the optimum is found using this algorithm.

3.2.2 Other dependencies

Implementation of the algorithm uses external procedure of random orthogonal matrix generating. Additionally, implementation relies on python numpy linear algebra functionality (matrices operation, eigendecomposition, etc.), on **pandas**[10] for loading data from a file, **scikit-learn**[8] for conducting preprocessing and measuring clustering algorithms, and on **matplotlib**[9] for visualization.

3.3 Description of general difficulties with problem which bear elaboration

Main challenge of the proposed implementation project, beside implementing Sub K-Means itself, is conducting experiments described in original paper over given sample datasets. Proper visualization of outputs, as a part of validating implementation, represent a significant difficulty.

3.4 Description of algorithm

3.4.1 Initialisation

Number of clusters *k* and dataset *D* of *d* dimensionality is accepted as algorithm input. *V* is random orthogonal matrix of *dxd* size. Dimensionality of clustered subspace is denoted by *m*. During

initialisation m is may be assigned a $d/2$ or $\sqrt{d}$. Next, dataset mean is calculated and denoted by μ_D Data set scatter matrix is denoted by S_D . Each cluster's mean is chosen randomly from dataset initially.

3.4.2 Clustering

SubKMeans proceeds by alternating between following steps until convergence:

Assignment step:

Assign each data point to the cluster with minimal cost function value. Cost function between data point x and cluster mean μ_i is defined as following:

$$\left\| P_c^T V^T x - P_c^T V^T \mu_i \right\|^2 \quad (1)$$

where P_c is a projection matrix onto first *m* attributes.

Update step:

Calculate new mean for each of k clusters: average coordinates of all data point belonging to cluster are considered to be new cluster mean. Also, for each cluster scatter matrix S_i is updated accordingly.

Eigendecomposition:

Matrix V is updated by performing eigendecomposition of following expression:

$$(\sum_{i=1}^{k} S_i) - S_D$$

Number of dimensions of clustering subspace is assigned to number of negative eigenvalues.

Convergence:

After each iteration following function is evaluated:

$$J = \left[\sum_{i=1}^{k} \sum_{x \varepsilon C_i} \left\| P_c^T V^T x - P_c^T V^T \mu_i \right\|^2 \right] + \sum_{x \varepsilon D} \left\| P_N^T V^T x - P_N^T V^T \mu_D \right\|^2$$

where P_N is a projection matrix onto last *d-m* attributes. When value of cost function doesn't change significantly through two consecutive iterations, algorithm stops and outputs clustering labels for each datapoint and final rotation matrix V.

3.4.3 Optimisation

Careful analysis of the algorithm showed that there are some improvements that can be introduced into implementation.

Formula (1) in 3.4.2 is calculated for each datapoint per cluster and for each iteration, or simply $O(k \times N \times ITERATIONS)$ times (N is dataset size). But the part $P_c^T V^T \mu_i$ is same for each cluster during one fixed iteration. It is optimal to calculate it in the beginning of each iteration and use result for calculating cost function value. Another part $P_c^T V^T x$ is also can be calculated once per datapoint and can be used further for measuring "distance" to each cluster. Eventually, per iteration we calculate only subtraction per cluster per datapoint, reducing amount of resource demanding matrix multiplications. In other words, we achieve complexity of $O((k+N) \times ITERATIONS)$.

4. Results & Findings

4.1 Datasets

4.1.1 Synthetic Datasets

Authors of the paper create this dataset in order to emphasize the performance of their new algorithm. We first apply our implementation of SubKMeans on this synthetic dataset used in the paper. Synthetic dataset has 1,500 instances and has 5 dimensions with 3 classes.

4.1.2 Real-World Datasets

We conduct our experiments on all of the real-world datasets used in this paper such as Wine, Pendigits, Ecoli, Seeds, Soybean, Symbol, OliveOil and Plane from UCI[11] and UCR[12] repositories.

- Wine dataset consists of 178 instances with 13 dimensions and 3 classes. Wine dataset used in the paper has dimensionality of 9, which is different from our dataset.
- Pendigits dataset has 10,992 instances in total and has dimensionality of 16 with integer types. This dataset was created by collecting 250 samples from 44 writers. The samples written by 30 writers are used for training, cross-validation and writer dependent testing, and the digits written by the other 14 are used for writer independent testing. Since authors use only training dataset, we also run experiments with the training dataset in which there exist 7,494 instances.
- Ecoli dataset is composed of 336 instances in total with 7 dimensions and 8 classes. We excluded 3 classes which contain only 2 or 5 instances while the other 5 classes have at least 20 instances, resulting in a dataset having 327 instances and 5 classes which is the same as the data mentioned in the paper.
- Seeds datasets has 336 instances and has 8 dimensions. The examined group comprised kernels belonging to three different varieties of wheat: Kama, Rosa and Canadian, 70 elements each, randomly selected for the experiment.
- Soybean dataset is composed of 995 instances with 398 dimensions and 6 classes.
- Symbols dataset consists of 995 instances in total and has dimensionality of 398 with 6 classes. Thirteen people participated in this experiment. They were asked to copy the randomly appearing symbol as best they could. There were 3 possible symbols, each person contributed about 30 attempts. The data is the X-Axis motion in drawing the shape.
- OliveOil dataset is composed of 60 instances and has 570 dimensions with 4 classes. Food spectrographs are used in chemometrics to classify food types.
- Plane dataset has 210 instances with a dimensionality of 144 and contains 7 classes.

4.1.3 Another Interesting Dataset

And additional interesting data sets including dancing stick figures from DSF repository[13], in which multiple clusters are intertwined in similar locations of space and have its own shape of figures are tested. This dataset consists of 9 basic stick figures with 9 classes from 1 to 9 and was built with 900 samples by randomly introducing noise. And space shuttle dataset from UCI is also used to check the relationship between the number of instances and the performance.

- Dancing Stick Figures dataset has 900 instances with a dimensionality of 400 and contains 9 classes.
- Space Shuttle dataset is used for experiments with 20000 instances with 9 dimensions and 7 classes.

4.2 Metrics

Experiments is held based on different evaluation metrics for clustering with ground truth with normalized mutual information score, Fowlkes-Mallows score and Silhouette Coefficient score. Mutual information of two random variables is a measure of the mutual dependence between the two variables. And normalized mutual information is a mutual information normalized by a certain variable entropy with normalized variants provided by the coefficients of constraint, uncertainty coefficient or proficiency. Fowlkes-Mallows score is a potential measure to be used with the ground truth class assignments of the samples. And Silhouette Coefficient is an evaluation metric which perform well even without the ground truth labelled data. For our experiment, we use Normalized Mutual Information, Fowlkes-Mallows and Silhouette Coefficients as metrics for measuring overall qualities of clustering algorithms.

Normalized Mutual Information score (NMI score)

Given the knowledge of the ground truth class label and our clustering algorithm predicted class label, the Mutual Information is a function that measures the agreement of the two assignments with ignoring permutations. Two different normalized versions of this measure are available, Normalized Mutual Information (NMI) and Adjusted Mutual Information (AMI), scaling the results between 0 (no mutual information) and 1 (perfect correlation). AMI is proposed recently and is normalized against chance but NMI is more often used in the literature.

Fowlkes-Mallows score (FMI score)

The Fowlkes-Mallows score can be used when the ground truth class assignments of the samples is known. And the score FMI is defined as the geometric mean of the pairwise precision and recall:

$$FMI = \sqrt{\frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}}$$

The score ranges from 0 to 1 and 1 means the highest value indicating a good similarity between two clusters, where T/F is True/False, P/N is Positive/Negative and TP is the number of True Positive case which is the number of points that have ‘True’ label ground truth and predicted as a ‘True’.

Silhouette Coefficient

The Silhouette Coefficient is an perfect option as an evaluation to be considered when the ground truth labels are not known, since evaluation must be performed using the model itself. A higher Silhouette Coefficient score relates to a model with better defined clusters and it can be defined for each sample and composed of two scores:

a: The mean distance between a sample and all other points in the same class.

b: The mean distance between a sample and all other points in the next nearest cluster.

The Silhouette Coefficient s for a single sample is then given as:

$$s = \frac{\{b - a\}}{\{max(a, b)\}}$$

Unlike the above MI based scores, the score has a bounded rounge [-1, 1], -1 for incorrect clustering, +1 for highly dense clustering (well separated clusters) and around zero for overlapping clusters. And the fact the Silhouette Coefficient is generally higher for convex clusters than other concepts of clusters, such as density based clusters (i.e. DBSCAN) needs to be considered.

Precision

Precision and Recall are the measurement based on an understanding of relevance among instances. And Precision is the fraction of relevant instances among the retrieved instances in information retrieval field generally while recall is the amount of retrieved one over the total amount of relevant instances. Sometimes these two measures are used together as the F-measure though, precision, recall, accuracy with RoC curve is the general measurement for a system. To use the concept in clustering problem, unlike NMI, precision needs more time as much as the number of permutations of n distinct clusters.

The score can be shown as below, where T/F is True/False, P/N is Positive/Negative and TP is the number of True Positive case which is the number of points that have 'True' label ground truth and predicted as a 'True'.

$$Precision = \sqrt{\frac{TP}{TP + FP}}$$

4.3 Experiments

We conducted experiments with SubKMeans' implementation measuring discussed metrics. Time was measured on the machine running OS MS Windows 10 on Intel Core i7-7700HQ@2.80GHz and 16GB RAM. We ran experiments 40 times, average metrics are presented in Table 1. Expected NMI is NMI achieved by authors of original paper. In general, our implementation achieves almost same results. We suppose that insignificant deviations may possibly come from various randomness in algorithm (initial cluster means are chosen by random, rotation matrix is also initiated randomly).

	Wine	Pendigits	Ecoli	Seeds	Soybean	Symbol	oliveoil	Plane	Stick Figures
NMI Expected	0.88	0.70	0.68	0.74	1.00	0.79	0.75	0.89	---
NMI (Our Implementation)	0.89	0.69	0.62	0.74	**1.00**	0.79	0.75	0.82	**1.0**
FMI	0.93	0.60	0.67	0.86	0.89	071	0.81	0.72	0.82
SC	0.28	0.28	0.27	0.40	0.34	0.45	0.33	0.45	0.27
Time (per one run), s	0.04	3.3	0.065	0.2375	0.02	3.38	0.08	0.19	0.585
Time (per 40 runs), s	1.60	132	2.60	9.5	0.80	135.20	3.20	7.60	23.40

Table 1. Evaluation of **SubKMeans (**our implementation**)**

In original paper synthetic dataset is discussed, on which PCA fails to reveal clear clustering structure. We run our implementation to get same visual results as ones presented in paper (Figure 1). Clusters are clearly visible on scatter plot (highlighted in red).

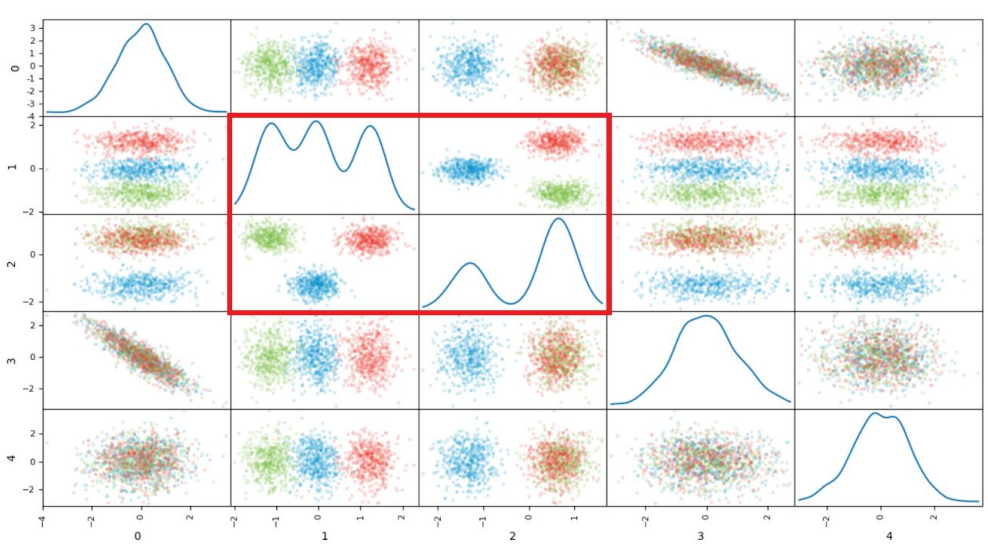


Figure 1. Visual results of clustering an artificial dataset

For each dataset, 7 different algorithm experiments were also conducted (PCA-Kmeans, ICA-Kmeans, LDA-Kmeans, ORCLUS, RBF-Kmeans, ACS-Kmeans and SCS-Kmeans), and each algorithm performance with three evaluation methods is summarised in each Table 2, 3, 4 and 5, respectively.

NMI	Wine	Pendigits	Ecoli	Seeds	Soybean	Symbol	Oliveoil	Plane	Stick fig	Shuttle
PCA-Kmeans	0.87	0.71	**0.66**	**0.74**	**1.00**	**0.78**	**0.75**	**0.91**	0.66	0.41
ICA-Kmeans	0.88	**0.69**	0.60	0.67	0.85	0.76	0.56	0.91	**1.00**	**0.30**
LDA-Kmeans	**1.00**	**0.81**	**0.72**	**0.87**	**1.00**	**0.94**	**1.00**	**1.00**	**0.97**	**0.47**
ORCLUS	0.88	0.65	0.69	0.73	1.00	X	X	X	X	X
RBF-Kmeans	0.01	0.00	0.66	0.65	0.13	0.04	0.69	0.24	0.02	0.00
ACS-Kmeans	**0.88**	0.68	0.66	0.72	**0.90**	0.77	0.73	0.83	0.66	0.31
SCS-Kmeans	0.81	0.51	0.65	0.59	0.72	0.71	0.67	0.83	0.33	0.20

Table 2. NMI scores

FMI	Wine	Pendigits	Ecoli	Seeds	Soybean	Symbol	Oliveoil	Plane	Stick fig	Shuttle
PCA-Kmeans	0.93	0.62	0.80	0.86	**1.00**	0.72	0.81	0.85	0.47	0.61
ICA-Kmeans	**0.93**	0.61	0.76	0.80	0.78	0.72	0.64	0.84	**1.00**	**0.56**
LDA-Kmeans	**1.00**	**0.79**	**0.83**	**0.94**	**1.00**	**0.94**	**1.00**	**1.00**	**0.93**	**0.53**
ORCLUS	0.93	0.56	0.82	0.85	**1.00**	X	X	X	X	X
RBF-Kmeans	0.34	0.10	0.78	0.79	0.29	0.19	0.79	0.25	0.11	0.30
ACS-Kmeans	0.93	0.60	0.80	0.84	0.87	0.71	0.81	0.75	0.47	0.51
SCS-Kmeans	0.88	0.40	0.75	0.73	0.66	0.63	0.77	0.75	0.29	0.40

Table 3. FMI scores

Silhouette	Wine	Pendigits	Ecoli	Seeds	Soybean	Symbol	Oliveoil	Plane	Stick fig	Shuttle
PCA-Kmeans	**0.45**	**0.62**	**0.39**	**0.86**	**0.63**	**0.72**	**0.81**	**0.60**	**0.46**	0.38
ICA-Kmeans	0.40	0.61	0.30	0.80	**1.00**	0.72	0.64	0.52	0.23	0.36
LDA-Kmeans	**0.66**	**0.79**	0.37	**0.94**	0.56	**0.94**	**1.00**	**0.93**	**0.88**	**0.26**
ORCLUS	0.28	0.28	**0.41**	0.41	0.36	X	X	X	X	X
RBF-Kmeans	0.01	0.01	0.30	0.10	0.02	0.01	0.34	0.03	0.01	0.01
ACS-Kmeans	0.30	0.25	0.36	0.40	0.21	0.43	0.36	0.37	0.29	0.26
SCS-Kmeans	0.30	0.15	0.33	0.40	0.28	0.15	0.37	0.40	0.03	0.30

Table 4. Silhouette

In Table 2 and 3, LDA Kmeans outperformed any other Kmeans algorithms generally (6 out of 9 datasets gave over 0.9 NMI score), followed by PCA Kmeans spreading out data with dimensionality reduction. While datasets which has large number of dimension compared to the number of instance, such as wine and soybean, generally performed better than other datasets, RBF Kmeans was well performed with Ecoli and Seeds. And ACS and SKE Kmeans which often used to computer vision, with their additive kernel, were greater than RBF Kmeans.

Table 4 shows how much it is important to choose the right evaluation method for each algorithm with silhouette's poor outcome compared to NMI and FMI. Unlike the phenomenon mostly LDA (Figure 2) by far outperformed other algorithms, with silhouette PCA was the best algorithm for every datasets, closely followed by LDA Kmeans.

By and large, Kernel methods are great for the time and high dimensional nonlinear data, but it seems it still needs experts' insights to choose the type of kernel methods for an tremendous achievement. This can be advantage and disadvantage at the same time. And the approximate feature

map provided by each Chi-squared kernel can be combined with the other approximate feature map. It may consider more data itself robust traits and provide by far amazing performance.

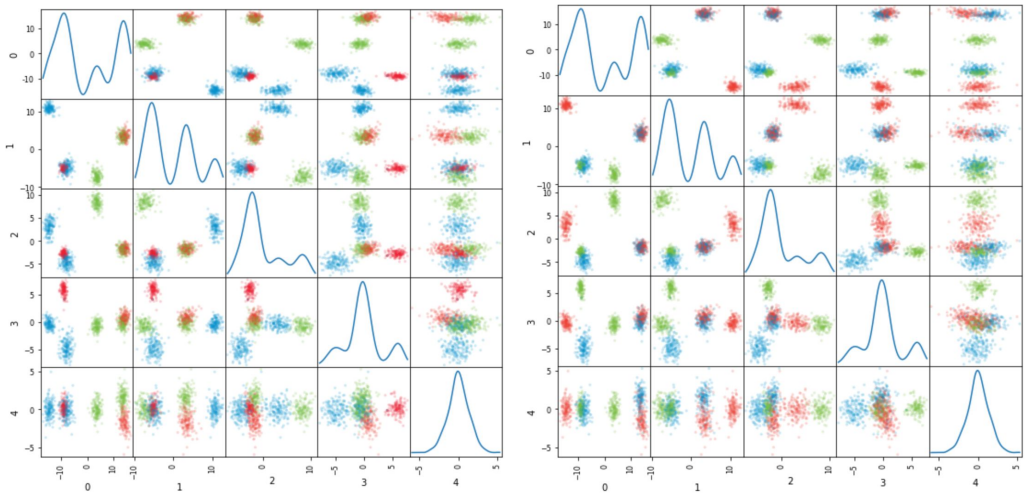
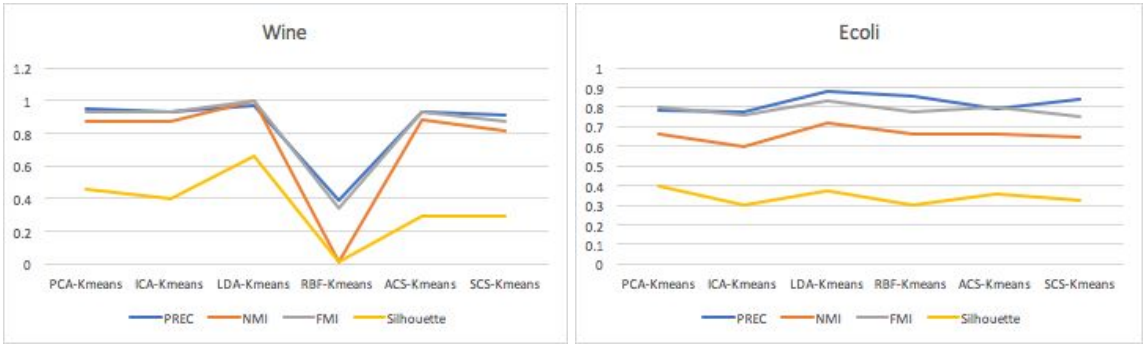


Figure 2. Symbols data with LDA prediction (Left) and ground truth (Right)

Precision	Wine	Ecoli	Seeds	Soybean	Symbol	Oliveoil
Sub-Kmeans	0.9616	0.7791	0.9310	0.9642	0.6735	0.8523
PCA-Kmeans	0.9535	0.7845	0.9200	0.9785	0.7190	0.8850
ICA-Kmeans	0.9285	0.7730	0.7375	1.0000	0.6995	0.7890
LDA-Kmeans	0.9730	0.8805	0.8755		0.9565	0.9700
RBF-Kmeans	0.3895	0.8565	0.7770	0.4255	0.2800	0.8715
ACS-Kmeans	0.9325	0.7905	0.8810	0.7600	0.6895	0.8790
SCS-Kmeans	0.9110	0.8360	0.8020	0.9570	0.5100	0.8900

Table 5. Precision



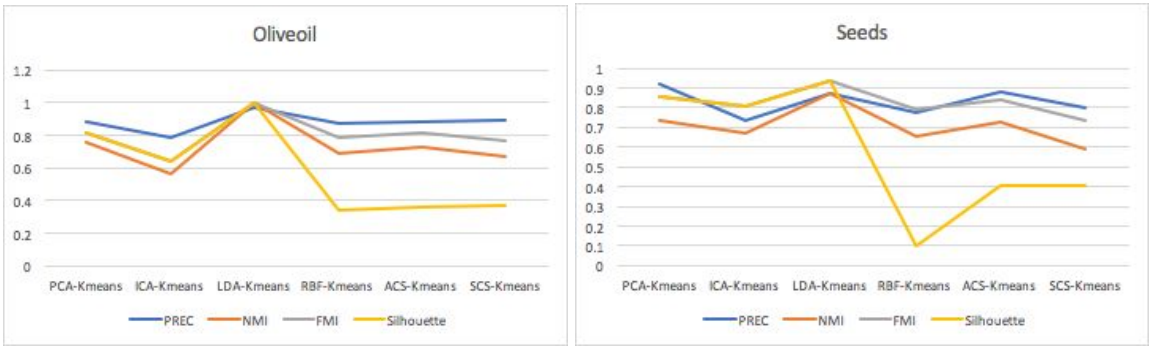


Figure 3. Evaluation methods Comparison

Table 5 describes the precision of major 7 algorithms with 6 datasets and figure 3 shows the similarity of pattern between precision and other evaluation methods. Mostly, the results of evaluation methods have the similar pattern, but different scale one. Therefore, considering the computational complexity and stability, NMI can be the best option for clustering performance measurement.

Exec Time (s)	Wine	Pendigits	Ecoli	Seeds	Soybean	Symbol	Oliveoil	Plane	Stick fig	Shuttle
SubKMeans (Our Implementation)	1.60	132	2.60	9.5	0.80	135.20	3.20	7.60	23.40	21.5
PCA-Kmeans	0.55	22.31	1.35	0.83	0.50	1.88	0.63	0.83	0.93	19.02
ICA-Kmeans	0.85	25.51	1.55	1.15	0.52	2.17	0.64	1.01	3.35	19.39
LDA-Kmeans	0.73	17.01	1.30	0.65	0.33	1.50	0.47	0.53	0.69	17.96
ORCLUS	0.68	9.86	0.66	0.48	0.72	X	X	X	X	X
RBF-Kmeans	0.85	103.75	1.40	1.10	0.34	5.91	0.62	1.24	4.93	388.29
ACS-Kmeans	0.73	19.68	1.21	0.84	0.53	20.69	1.62	1.56	15.78	20.28
SCS-Kmeans	0.81	35.1	1.59	1.04	0.49	3.17	0.60	0.89	5.65	43.01

Table 6. Running time (40 runs)

We conducted running time experiments for all of algorithms on 10 datasets which have various dimensions and sizes of data points. We measured execution time of algorithms to run 40 times (see Table 6). Comparing to other algorithms, SubKMeans of our implementation shows overall worse performance in terms of running time. For datasets containing high number of dimensions such as Pendigits, Symbol, OliveOil, and Dancing Stick Figures, SubKMeans of our implementation takes more time than other datasets with small number of dimensions. SubKMeans yields bad performance in Seeds dataset which has small number of instances and dimensions in which SubKmeans takes long time to converge.

5. Discussion & Conclusion

First of all, We describe essential mathematical background knowledge to better understand how SubKMeans works and other clustering methods such as basic linear algebra and kernel method. K-means which is one of the most basic clustering algorithms and procedures of Implementation of SubKMeans are formally described.

We confirm that our implementation and author's SubKMeans shows almost same results by means of comparing the NMI score and visualization result of synthetic dataset that the author provide. In addition, We conduct experiments of 8 different clustering algorithms including SubKMeans on 10 datasets with different number of instances, classes, and dimensions in order to fully compare SubKMeans with existing clustering algorithms. The results of clustering algorithms are compared with 5 different metrics such as NMI score, FMI score, Silhouette coefficient, Precision, and running time.

For NMI score, FMI score, and Silhouette coefficient, SubKMeans cannot show highest performance but rather moderate performance among 8 clustering algorithms. For Precision, SubKMeans shows the highest precision in some datasets**.** Although SubKMeans performs relatively well in terms of quality of clustering, the running time evaluation shows that SubKMeans works worse than other clustering algorithms compared in our experiments.

6. References

[1] Ian Jolliffe. 2002. Principal component analysis. Wiley Online Library

[2] Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. Neural networks 13, 4 (2000), 411–430.

[3] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. science 290, 5500 (2000), 2319– 2323

[4] Wei Ye, Samuel Maurus, Nina Hubig, and Claudia Plant. “Generalized Independent Subspace Clustering.” ICDM 2016

[5] Cristianini, N. and Shawe-Taylor, J. “An introduction to support vector machines and other kernel-based learning methods.” 2000

[6] Pelleg, Dan, and Andrew W. Moore. "X-means: Extending K-means with Efficient Estimation of the Number of Clusters." ICML. Vol. 1. 2000.

[7] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: spectral clustering and normalized cuts. SIGKDD 2004

[8] Pedregosa et al., Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011.

[9] Hunter, J. D., Matplotlib: A 2D graphics environment, Computing In Science & Engineering 9, 3(2007), 90-95.

[10] Wes McKinney, Data Structures for Statistical Computing in Python, 9th Python in Science Conference. 51-56. 2010.

[11] M. Lichman. UCI machine learning repository, 2013.

[12] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery , Online First, 2016.

[13] Stephan G ¨unnemann, Ines F ¨arber, Matthias R ¨udiger, and Thomas Seidl. Smvc: Semi-supervised multi-view clustering in subspace projections. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’14, pages 253–262, New York, NY, USA, 2014. ACM.