Journal of
open research software

Lavielle, M and Waggoner, P D 2020 Exploring and Comparing Unsupervised Clustering Algorithms. *Journal of Open Research Software,* 8: 21. DOI: https://doi.org/10.5334/jors.269

SOFTWARE METAPAPER

Exploring and Comparing Unsupervised Clustering Algorithms

Marc Lavielle[1] and Philip D. Waggoner[2]

[1] Center of Applied Mathematics, Inria and Ecole Polytechnique, FR

[2] Computational Social Science, University of Chicago, US

Corresponding author: Philip D. Waggoner (pdwaggoner@uchicago.edu)

One of the most widely used approaches to explore and understand non-random structure in data in a largely assumption-free manner is clustering. In this paper, we detail two original Shiny apps written in R, openly developed at Github, and archived at Zenodo, for exploring and comparing major unsupervised algorithms for clustering applications: k-means and Gaussian mixture models via Expectation-Maximization. The first app leverages simulated data and the second uses Fisher's Iris data set to visually and numerically compare the clustering algorithms using data familiar to many applied researchers. In addition to being valuable tools for comparing these clustering techniques, the open source architecture of our Shiny apps allows for wide engagement and extension by the broader open science community, such as including different data sets and algorithms.

Keywords: unsupervised clustering; Gaussian mixture models; k-means; EM; Shiny; R

(1) Overview

Introduction

In the data science age, the ability to effectively explore and understand data is a central task. In service of this task, unsupervised machine learning provides many useful tools for exploring and understanding data in a mostly assumption-free manner [11]. One of the most widely used approaches for doing so is clustering, which offer researchers sophisticated algorithms for grouping observations to maximize similarity in feature space, and in so doing, uncover natural, non-random structure in data [1]. Among the most commonly used techniques are k-means and Gaussian mixture models for clustering.

K-means is a clustering technique that groups observations in a shared space based on spatial similarity, in pursuit of *intra*-cluster homogeneity and *inter*-cluster heterogeneity [2]. The goal is to minimize within cluster sums of squares based on distances to a computed cluster centroid, which in the k-means case, is a computed cluster mean.[1]

The goal of using Gaussian mixture models for clustering applications is similar to the k-means goal of maximizing similarity across observations in a shared space. Yet the key difference is that cluster assignment in the mixture model case is determined probabilistically [3]. Observations that have similar probabilities of belonging to the same component (or "cluster") are grouped together, but with a degree of uncertainty. A further difference is the presence and use of probabilities, making most mixture models parametric. As such, the Expectation-Maximization (EM) algorithm is an iterative algorithm that is most often used to estimate the parameters, which constrain the clustering procedure [10]. Once each of the components in the mixed space are identified, clustering is carried out by assigning each observation to the most likely component in the mixture, which is determined by maximizing the posterior probabilities of each observation belonging to each component.

In this paper, we present two original Shiny apps written in the R language [4], that allow researchers in any field to explore, understand, and directly compare clustering results from these widely used clustering algorithms. The first Shiny app is based on simulated data to ease the user into the logic of clustering, and the second leverages the Iris data, and is expanded to also include observation-level numeric output (e.g., cluster assignments) in addition visual output. Both apps leverage ggplot2 graphics [5], which contribute to clean, simple user interfaces to contribute to deeper understanding.

While our main goal is to encourage exposure to and interaction with these widely used clustering algorithms, our Shiny apps can easily be extended in future projects by, for example, including other data sets as well as other algorithms to widen to comparability. Indeed, as the apps are developed openly, we encourage such contributions. Avenues for contribution are discussed below as well as at the corresponding Github repository under the "Contribution" section in the README.

Implementation and architecture

The software for each app was implemented by programming a user interface (UI) and a server, as necessary component parts of a Shiny app. Upon creation of the UI and server, each app was consolidated into a single script, and then deployed at the corresponding author's Shiny host site, pdwaggoner.shinyapps.io. More details on deployment are available in the following "Availability" section.

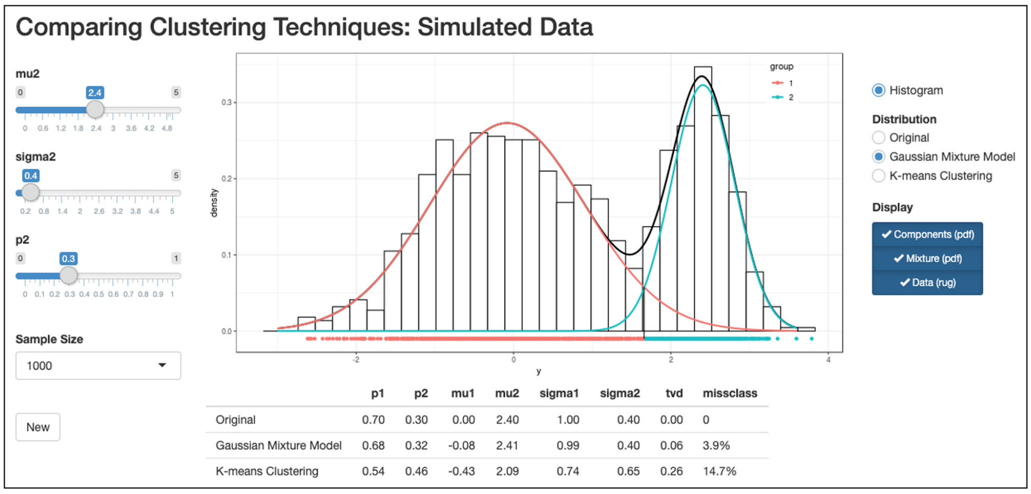
To get a clearer sense of the architecture of each app, consider the screenshots of the launched apps below in **Figures 1–3**. **Figure 1** shows the first clustering app using simulated data. **Figure 2** shows the visual component of the second clustering app using the Iris data. **Figure 3** shows the numeric output from the second clustering app also using the Iris data.

In **Figure 1**, the raw data densities are displayed along with an optional histogram for another view, as well as the computed cluster densities. The numeric output just below the visual output compares across each technique as well as the original raw inputs. P1 and P2 correspond with the input features, mu1 and mu2 correspond with the density means, and sigma1 and sigma2 correspond with the variance of each component/cluster. Users are simply able to slide the nodes in the left-hand panel to see the densities and clusters change in real time. Users also have a number of display options in the lower right, including the probability density functions (components), as well as a rug plot of the observations.

Further, in the first app in **Figure 1**, data are drawn from a mixture of two univariate Gaussian distributions. The first distribution has mean 0 and standard deviation 1 while the mean mu2, the standard deviation sigma2 and the proportion p2 of the second distribution are defined by the user. The sample size can also be altered, ranging from 50 to 2000.

As users are able to determine the aspects included on the visual rendering of the patterns (e.g., including a scatter plot of the data, a histogram, the probability distribution density of each component), the objective of this app is to show that k-means clustering is well adapted when the two distributions are quite similar, except for a difference in the mean. On the other hand, Gaussian mixtures are better suited when the distributions have quite different standard deviations and/or proportions.

Turning to the second app in **Figure 2**, we use the famous (Fisher's or Anderson's) Iris data set, which gives the measurements in centimetres of the variables sepal length and width and petal length and width, respectively, for


Comparing Clustering Techniques: Simulated Data

mu2: 2.4

sigma2: 0.4

p2: 0.3

Sample Size: 1000

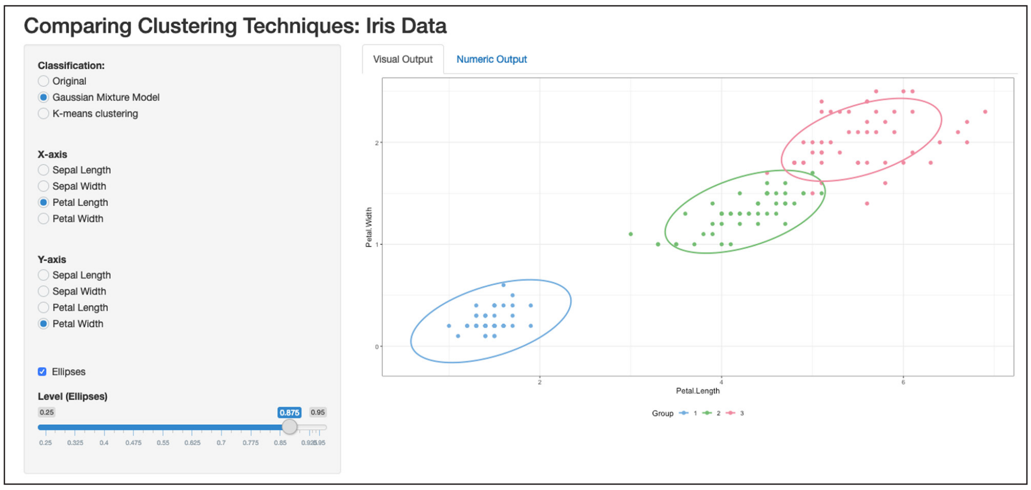
[✓ Histogram]

Distribution: Original [✓ Gaussian Mixture Model] K-means Clustering

Display: ✓ Components (pdf) ✓ Mixture (pdf) ✓ Data (rug)

	p1	p2	mu1	mu2	sigma1	sigma2	tvd	missclass
Original	0.70	0.30	0.00	2.40	1.00	0.40	0.00	0
Gaussian Mixture Model	0.68	0.32	-0.08	2.41	0.99	0.40	0.06	3.9%
K-means Clustering	0.54	0.46	-0.43	2.09	0.74	0.65	0.26	14.7%

Figure 1: Clustering App using Simulated Data.


Comparing Clustering Techniques: Iris Data

Classification: Original [✓ Gaussian Mixture Model] K-means clustering

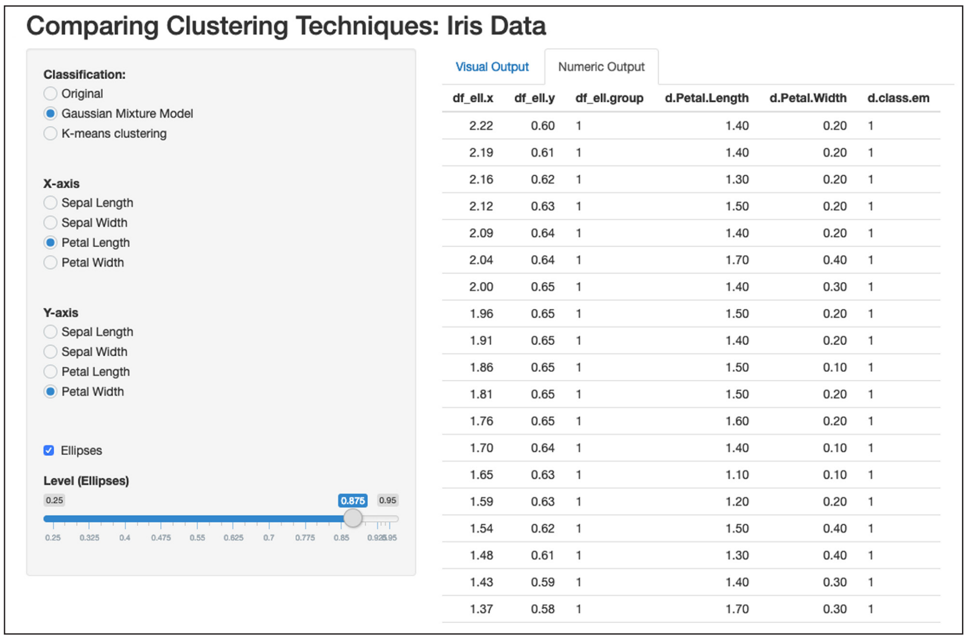
X-axis: Sepal Length Sepal Width [✓ Petal Length] Petal Width

Y-axis: Sepal Length Sepal Width Petal Length [✓ Petal Width]

[✓ Ellipses]

Level (Ellipses): 0.875

Figure 2: Visual Output of the Second Clustering App using Iris Data.


Comparing Clustering Techniques: Iris Data

Classification:
- ○ Original
- ● Gaussian Mixture Model
- ○ K-means clustering

X-axis
- ○ Sepal Length
- ○ Sepal Width
- ● Petal Length
- ○ Petal Width

Y-axis
- ○ Sepal Length
- ○ Sepal Width
- ○ Petal Length
- ● Petal Width

☑ Ellipses

Level (Ellipses): 0.875 (range 0.25 – 0.95)

Visual Output | Numeric Output

df_ell.x	df_ell.y	df_ell.group	d.Petal.Length	d.Petal.Width	d.class.em
2.22	0.60	1	1.40	0.20	1
2.19	0.61	1	1.40	0.20	1
2.16	0.62	1	1.30	0.20	1
2.12	0.63	1	1.50	0.20	1
2.09	0.64	1	1.40	0.20	1
2.04	0.64	1	1.70	0.40	1
2.00	0.65	1	1.40	0.30	1
1.96	0.65	1	1.50	0.20	1
1.91	0.65	1	1.40	0.20	1
1.86	0.65	1	1.50	0.10	1
1.81	0.65	1	1.50	0.20	1
1.76	0.65	1	1.60	0.20	1
1.70	0.64	1	1.40	0.10	1
1.65	0.63	1	1.10	0.10	1
1.59	0.63	1	1.20	0.20	1
1.54	0.62	1	1.50	0.40	1
1.48	0.61	1	1.30	0.40	1
1.43	0.59	1	1.40	0.30	1
1.37	0.58	1	1.70	0.30	1

Figure 3: Numeric Output of the Second Clustering App using Iris Data.

50 flowers from each of three species of iris: Iris setosa, versicolor, and virginica.

In the second app, users can select the variables to display on the X and Y axes, as well as the ellipses sizes, which is a 2D extension of a prediction interval for bivariate distributions. It is important to note that a mixture of multivariate Gaussian distributions that takes into account correlations between variables results in an excellent classification (only 2% of misclassified flowers). See http://sia.webpopix.org/mixtureModels.html for more details about the algorithms used in these apps.

Quality control

The apps have been thoroughly quality checked and tested locally. First, the UI's have been tested on the several widely used internet platforms including Firefox Quantum 65.0.2, Chrome 72.0.3626.121, Safari 12.0.3, and Opera 58.0.3135.107. No errors or problems occurred on any browser or system. The apps worked efficiently and as expected on all platforms.

Second, the code was locally tested using the shinytest package, which is a unit testing package for Shiny apps. This package tests for the following, which was copied and pasted from RStudio for consistency:[2]

1. "First is the test driver. This is the R process that coordinates the testing and controls the web browser. When working on creating tests interactively, this is the R process that you use.
2. Next is the Shiny process, also known as the server. This is the R process that runs the target Shiny application.
3. Finally, there is the web browser, also known as the client, which connects to the server. This is a headless web browser – one which renders the web page internally, but doesn't display the content to the screen (PhantomJS)."

Across all of these local tests for both apps, which are catalogued at the Github repository under the "tests" folder for the simulated data app (1) and "tests_iris" for the Iris data app (2), there were zero errors thrown, and all code and apps behaved as expected.

Finally, at the Github repository listed below, users can inspect all source code, unit testing results, as well as inspect the README for more details on each app as well as how to diagnose the results both visually and numerically (see the "Understanding the Output" subsection in the README).

(2) Availability

The software includes two Shiny apps that are hosted separately at unique Shiny app pages. The first app is hosted at https://pdwaggoner.shinyapps.io/App1-Simulated_Data/ and the second app is hosted at https://pdwaggoner.shinyapps.io/App2-Iris_Data/. The code is openly developed at a single Github repository: https://github.com/pdwaggoner/clustering_Shiny.

Operating system

The apps were originally built and launched on MacOS Mojave 10.14.3.

Programming language

The apps were originally written in R v3.5.2 and RStudio v1.2.1206.

Dependencies

shiny [4]; ggplot2 [5]; htmltools [6]; ellipse [7]; shinyWidgets [8]; kmeans (Base R); mixtools [9].

List of contributors

- Marc Lavielle, Center of Applied Mathematics, Inria & Ecole Polytechnique
- Philip D. Waggoner, Computational Social Science, University of Chicago

Software location

Archive

Name: Exploring and Comparing Unsupervised Machine Learning Clustering Algorithms
Persistent identifier: DOI: 10.5281/zenodo.2595293
Licence: MIT
Publisher: Philip D. Waggoner
Version published: v1
Date published: 15/03/19

Code repository

Name: clustering_Shiny
Identifier: https://github.com/pdwaggoner/clustering_Shiny
Licence: MIT
Date published: 15/03/19

Language

English

(3) Reuse potential

Our web apps are valuable for researchers, practitioners, and academics in any field that use machine learning and clustering techniques for data analysis, mining, and exploration. As we demonstrate in both apps, clustering algorithms are extremely effective in revealing underlying, non-random structures in data, whether simulated (first app) or observational (second app).

For those interested in contributing to these apps, which is highly encouraged, there are three main options to do so. First, users may open an issue ticket to suggest a change, report a bug, or even ask a question at the Github repository. Second, users may directly make a change by opening a pull request at the Github repository. Or finally, if less familiar with open source collaboration of this sort, users are welcome to directly reach out to either (or both) author(s) to make suggestions, report bugs, or request feature enhancements.

Further, there are many opportunities for extension given that these methods are not unique to or bound by a specific field or subfield. From biology and medicine to ecology and social science, unsupervised clustering algorithms are widely used and valuable for a variety of questions in a variety of domains. Thus, building on our software, researchers could reuse and extend either by adding additional clustering algorithms to expand comparison (e.g., hierarchical agglomerative clustering or k-medoids clustering), or by updating the apps to allow for uploading original datasets to visually, quickly, and efficiently explore how patterns and clusters emerge dependent on the algorithm selected, but in specific research contexts.

In sum, our apps provide an important starting place, not only for exploring and comparing unsupervised clustering algorithms, but also for offering a baseline for future researchers, academics, and practitioners to update and expand our tools to widen usage and application. In line with open source and open research, we encourage reuse cases and extensions of our work.

Notes

[1] This is distinct from the close relative, k-medoids clustering, which specifies a representative observation as the cluster centroid, rather than computing a cluster mean based on cluster members as in the k-means case. The two, though, are computationally similar.

[2] https://blog.rstudio.com/2018/10/18/shinytest-automated-testing-for-shiny-apps/.

Acknowledgements

These Shiny apps are based on apps by Marc Lavielle, which were originally used for teaching purposes at Lavielle's educational website "Statistics in Action" http://sia.webpopix.org/mixtureModels.html.

Competing Interests

The authors have no competing interests to declare.

References

1. **Jain, A K, Murty, M N** and **Flynn, P J** 1999 Data clustering: a review. *ACM computing surveys (CSUR),* 31(3): 264–323. DOI: https://doi.org/10.1145/331499.331504
2. **Hartigan, J A** and **Wong, M A** 1979 Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1): 100–108. DOI: https://doi.org/10.2307/2346830
3. **Banfield, J D** and **Raftery, A E** 1993 Model-based Gaussian and non-Gaussian clustering. *Biometrics*, 803–821. DOI: https://doi.org/10.2307/2532201
4. **Chang, W, Cheng, J, Allaire, J, Xie, Y** and **McPherson, J** 2015 Shiny: web application framework for R. *R package version 1.2.0.*
5. **Wickham, H** 2016 *ggplot2: Elegant graphics for data analysis*. Springer. DOI: https://doi.org/10.1007/978-3-319-24277-4
6. **Cheng, J** 2017 htmltools: Tools for HTML. *R package version 0.3.6.*
7. **Murdoch, D** and **Chow, E D** 2018 ellipse: Functions for drawing ellipses and ellipse-like confidence regions. *R package version 0.4.1.*
8. **Perrier, V, Meyer, F** and **Granjon, D** 2019 shinyWidgets: Custom inputs widgets for Shiny. *R package version 0.4.7.*
9. **Young, D, Benaglia, T, Chauveau, D** and **Hunter, D** 2017 mixtools: Tools for Analysing Finite Mixture Models. *R package version 1.1.0.*
10. **McLachlan, G J, Lee, S X** and **Rathnayake, S I** 2000 Finite mixture models. *Annual Review of Statistics and Its Application, no. 0.*
11. **Waggoner, P D** *Forthcoming Unsupervised Machine Learning for Clustering in Political and Social Research.* New York: Cambridge University Press.

How to cite this article: Lavielle, M and Waggoner, P D 2020 Exploring and Comparing Unsupervised Clustering Algorithms. *Journal of Open Research Software,* 8: 21. DOI: https://doi.org/10.5334/jors.269

Submitted: 19 March 2019 **Accepted:** 16 September 2020 **Published:** 07 October 2020