

Ensemble Metrics And Models For Density-Based Clustering

Thomas Charlon
Harvard Medical School

Abstract

The OPTICS k-Xi density-based clustering pipeline now integrates ensemble metrics which enable to increase the stability of the clusters and the reproducibility of the results on evaluation datasets. Up to now, OPTICS k-Xi computed several distance-based metrics but did not implement any way of merging together their results, it was up to the user to select the most relevant metric or to merge them together.

Recently, we applied the pipeline on natural language processing embeddings and the choice of the relevant metric revealed complex and the clusters were unstable. To alleviate this, we implemented ensemble metrics and models in the OPTICS k-Xi pipeline. Three other improvements have also been implemented: the selection of the cosine distance; a parameter to set a maximum cluster size; and a parameter to select a minimum number of clusters.

This vignette showcases and introduce the new developments of the **opticskxi** package, with a particular focus on ensemble metrics and models.

Keywords: Density-based clustering, hierarchical clustering, distance-based metrics, ensemble models.

1. New developments

1.1. Ensemble metrics and models

Unsupervised clustering methods usually rely on distance-based metrics such as the between-within ratio, the Dunn index, the average silhouette width, etc. The `fpc` package provides several distance-based metrics and depending on the use case some may be preferred to others, *e.g.* the Dunn index will favor a minimal distance between all clusters, while the between-within ratio will favor an average ratio between the size of the clusters and the distance between all clusters.

In some applications however, it is unclear to the analyst which metric should be preferred, especially in research and knowledge discovery applications in which the clusters are yet to be characterized, understood, and investigated. Thus many real-world clustering studies will include several distance-based metrics, although one challenge is that they are hardly comparable.

Ensemble models are a family of models that merge results from different clustering models using voting mechanisms. One commonly used method is to have models each vote for the best result, then sum up the votes and select the result with most votes. Thus the results are compared by summing their ranks in each distance-based metric, rather than the actual value of the metric.

This voting mechanism can be implemented in several ways and several questions can be considered: should each clustering model vote just for the best result or should all results be ranked? should similar metrics be incorporated or should each metric be measuring something very different (*e.g.* the between-within ratio and the average silhouette width will often vote for the same model while the Dunn index produces very different results)?

1.2. Selection of cosine distance

Previously, while OPTICS k-Xi enabled to choose from 4 distances, the distance-based metrics were only computed using the euclidean distance, as one distance must be chosen to compute the distance-based metrics to compare clustering models that make use of different distances. It is now possible to choose to perform the distance-based metrics on the cosine distance using the `metrics_dist` parameter.

1.3. Maximum cluster size

One of the limitations of using distance-base metrics with hierarchical density-based clustering is that since some clusters may be nested within others, the minimal distance between all clusters becomes null, and many distance-based metrics rely on this concept of cluster separation. We observed that the main issue came from when points with high OPTICS distance are clustered together in one large cluster that surrounds all others, and that most often having a few small nested clusters would not impact the metrics significantly, and thus we implemented a restriction on the maximum size of a cluster with the `maz_size_ratio` parameter. Another solution that could be investigated in the future would be to incorporate the hierarchical structure within the distance metrics.

1.4. Minimum number of clusters

While the OPTICS k-Xi method enables to specify the number of largest distance differences to use to determine clusters, this does not translate directly as the number of clusters, as several parameters interact to produce the final number of clusters (*e.g.* the minimum number of points to produce a neighborhood) and a specific number of clusters will not be reachable in every dataset. When we have prior knowledge about the dataset we may know that we are expecting at least a specific number of clusters, however the models with small number of clusters may have artificially high distance-based metrics compared to models with more clusters.

Thus we may want to exclude the models with a below threshold number of clusters, and for that we implemented the `n_min_clusters` parameter to set a minimum number of clusters and exclude the models that do not meet the criteria.

2. Example dataset

2.1. Word2vec embeddings of open-access publications

The dataset investigated is the result of applying word2vec on a large corpora of 1,700 mental health related scientific publications. Word2vec emerged in the mid 2010s as a powerful method for natural language processing and for discovering similarities between concepts in unstructured text (Mikolov 2013). Recent developments in large language models (LLMs) now produce far better results than word2vec on a number of benchmarks, however word2vec still proves useful, especially when performed on study-specific data and combined with pre-trained LLMs.

Europe PMC maintains a FTP site enabling to download open-access publications, which is introduced in the vignette of the **tidypmc** package. After downloading the publications locally, the full-texts of 1,700 publications were pre-processed by removing irrelevant sections (*e.g.* supplementary data), transforming all text to lowercase, and removing partly the punctuation (but keeping *e.g.* dashes).

The word2vec embeddings were performed with the **text2vec** package and applied on the pre-processed input text with an embedding dimension parameter of 100. An embedding matrix of more than 30,000 words in rows and 100 embedding dimensions columns was obtained, in which each word is represented by a vector of 100 numeric values. Our objective here is to discover a landscape of the global knowledge contained in the publications, *i.e.* ontology learning. For that, we want to discover groups of words, *i.e.* related concepts, within the matrix.

Additional details about the dataset were provided at R in Medicine 2024 and the video is available online at <https://www.youtube.com/watch?v=sj7qYS8oRyc>.

2.2. Manual evidence of clusters

Single word queries

Using the embedding matrix, one can select a word, *e.g.* "medication", fetch the corresponding vector representation, and compare it to all other word representations and return the 50 closest matches. In this dataset, "medication" will return terms "antipsychotic" and "antidepressant", which are examples of mental health medications. This is one example of a group of related concepts, which we call clusters. Similarly, the word "therapy" will return "cbt" and "dbt", acronyms of cognitive behavioral therapy and dialectical behavior therapy.

Vector operations

Embedding models are well suited to perform vector operations. A well-known example illustrates this: on an embedding matrix trained on general text, one can take the embedding vector of the word "Paris", subtract the one of "France", and add the one of "Germany", to produce a list of closest matches (*e.g.* with the cosine similarity) which will include the word "Berlin", thus indicating the capability of the model to understand language semantics. Written as a vector operation, this becomes "Paris" - "France" + "Germany" = "Berlin", and in natural language one could say "Paris is to France what Berlin is to Germany". One particularly interesting aspect of this example is that the top matches will also include other capitals as "Moscow", thus suggesting the evidence of a "capitals subspace".

Our exploration of single word queries and vector operations enabled us to find lists of concepts that seemingly constituted clusters. To confirm our hypothesis that these constituted objective clusters, we applied the unsupervised density-based clustering method OPTICS k-Xi.

2.3. Word2vec embeddings subset

We cannot apply OPTICS k-Xi on the global embedding matrix as the number of features is too large. We thus first compiled a list of terms based on our manual exploration for which we expected that many would fall into well-separated clusters. We chose 23 single word queries and/or vector operations, each combining additions and/or subtractions from 1 to 4 words, and for each of them selected the 50 closest words (*i.e.* with highest cosine similarity). This produced a list of 831 unique words, and we subset our embedding matrix to those words and call OPTICS k-Xi on it. This embedding matrix subset is provided with the **opticskxi** package.

3. Density-based clustering with ensemble metrics

3.1. Individual metrics

To demonstrate the advantages of ensemble metrics, we first show the limitations we encounter when using single metrics and compare the results with the ensemble metrics approach. We start by calling the OPTICS k-Xi pipeline with the newly implemented parameters (*metrics_dist*, *max_size_ratio*, *n_min_clusters*) and plot the default metric (average silhouette width) (Figure 1). We newly implemented the selection of the cosine distance, a parameter to set a maximum cluster size, and a parameter to set a minimum number of clusters.

```
R> library('opticskxi')
R> data('m_psych_embeds')
R> set.seed(0)
R> df_params = expand.grid(n_xi = 8:15, pts = c(15, 20, 25, 30),
+                         dist = "cosine", dim_red = "ICA",
+                         n_dimred_comp = c(10, 15, 20, 25))
R> df_kxi = opticskxi_pipeline(m_psych_embeds, df_params,
+                             metrics_dist = 'cosine',
+                             max_size_ratio = 0.15, n_min_clusters = 5,
+                             n_cores = 1)
R> plot(gtable_kxi_profiles(df_kxi))
```

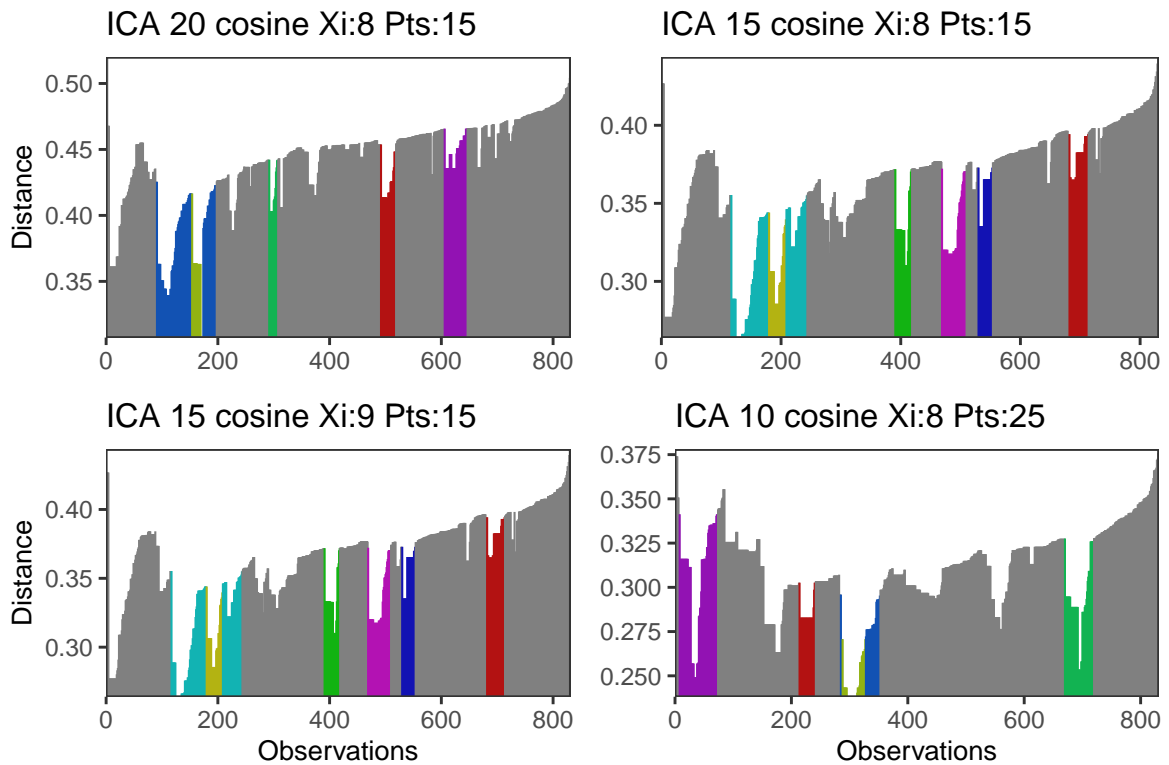


Figure 1: OPTICS k-Xi best 4 models for average silhouette width, ordered by columns then rows. The OPTICS distance profiles for the top models are quite different. The top profile (upper left) slightly resembles the characteristic logarithmic profile of mediocre models indicating every points are very distant from each other and the clustering model is not optimal. Although some clusters were found, they are very small.

Let's now have a look at two other common metrics: the between-within ratio (Figure 2) and the Dunn index (Figure 3).

```
R> plot(gtable_kxi_profiles(df_kxi, metric = 'bw.ratio'))
```

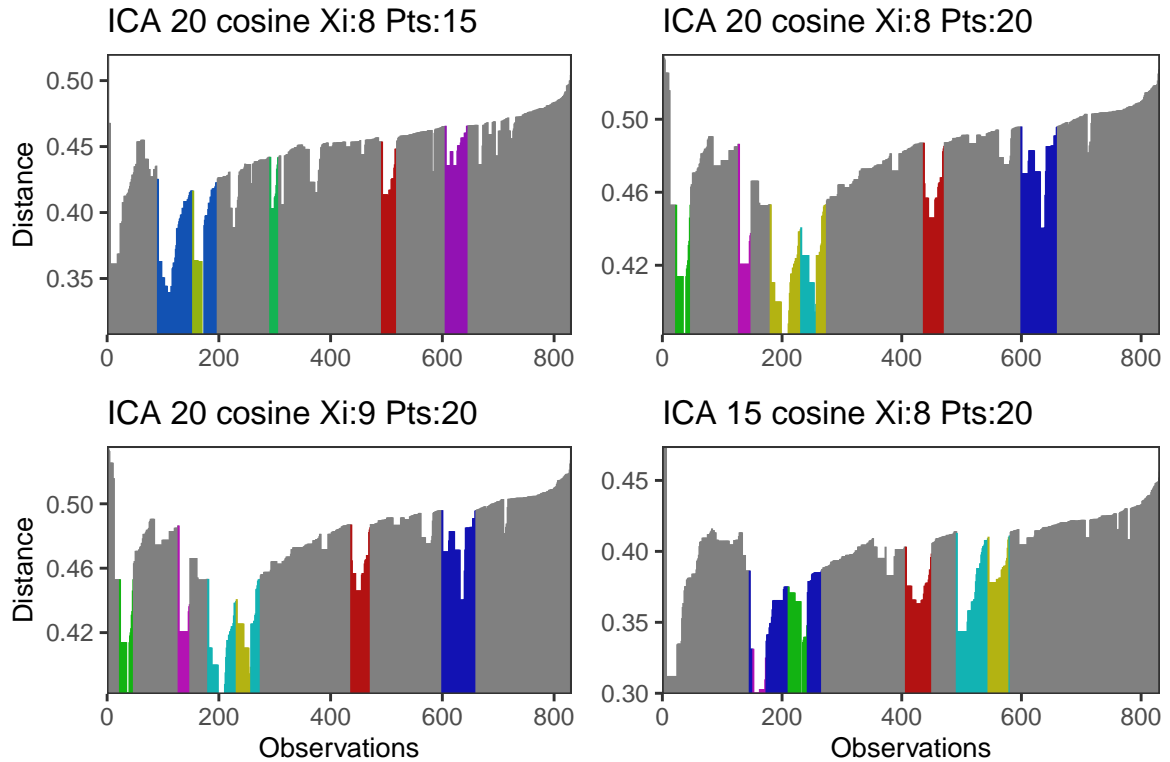


Figure 2: OPTICS k-Xi best 4 models for between-within ratio, ordered by columns then rows. The top model is the same as for average silhouette width, which is expected since the metrics are similar.

```
R> plot(gtable_kxi_profiles(df_kxi, metric = 'dunn'))
```

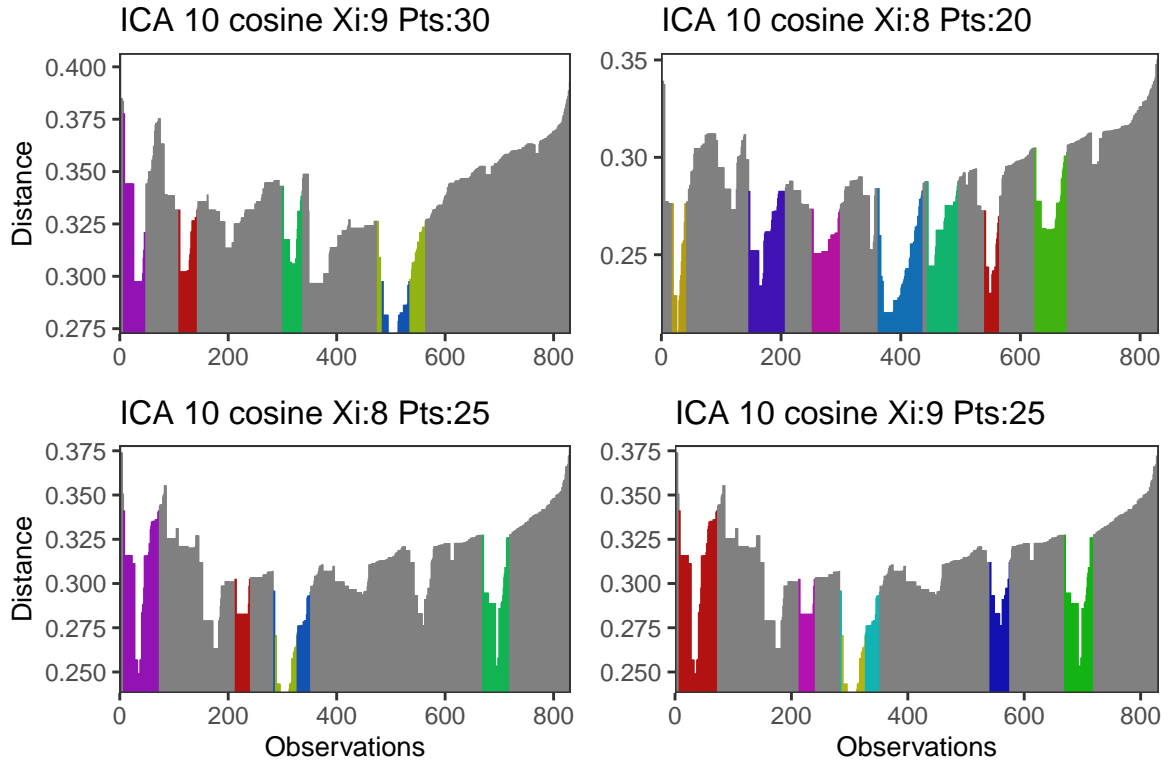


Figure 3: OPTICS k-Xi best 4 models for Dunn index, ordered by columns then rows. The second model might be more interesting to investigate, as we obtain more clusters and less points fall into the noise (in grey).

We can also plot the metrics values (Figure 4).

```
R> plot(ggplot_kxi_metrics(df_kxi, n = 15,
+                          metric = c("avg.silwidth", "bw.ratio", "dunn")))
```

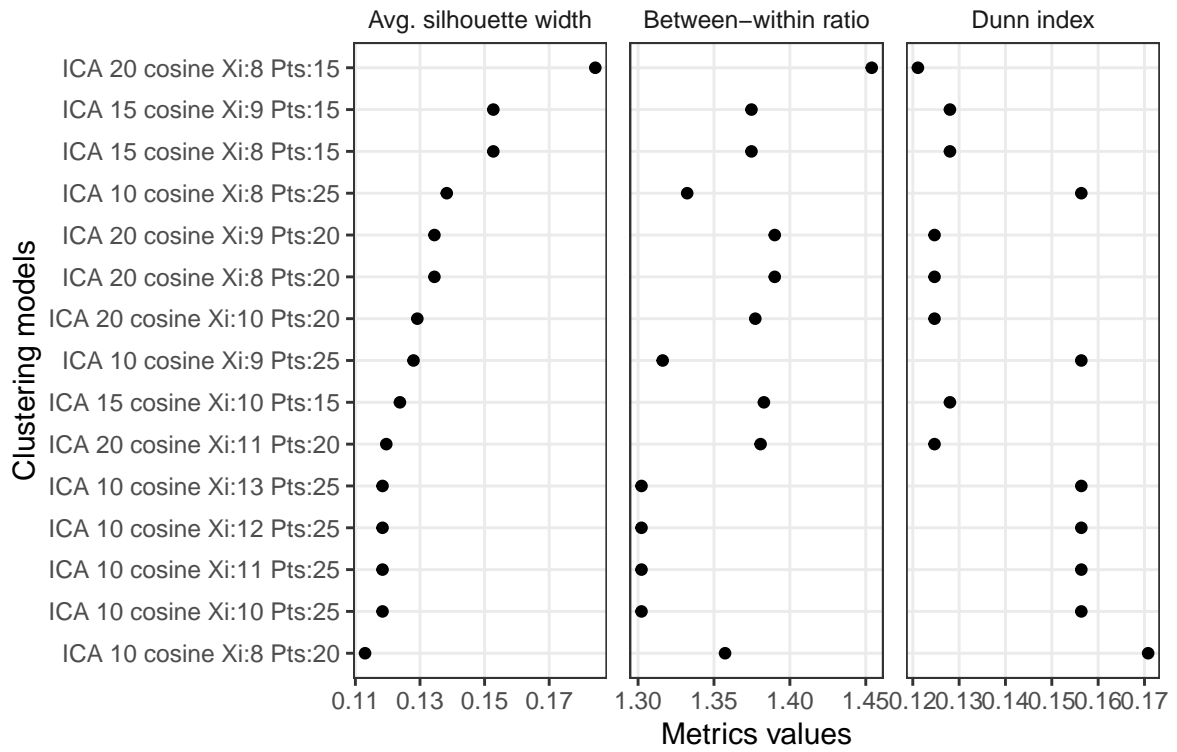


Figure 4: Numeric values of metrics for the top models by average silhouette width. Average silhouette width and between-within ratio have mostly similar results, while the Dunn index results are very different.

3.2. Ensemble metrics and models

The ensemble metrics and models have been developed as two nested modules with their own sets of parameters.

Summing of thresholded ranks

The function `ensemble_metrics` is the most inner one and will rank the metrics pre-computed by the OPTICS k-Xi pipeline. Here the parameters are:

- `n_top` Threshold of number of models to rank
- `df_params` The models dataframe, output of `opticskxi_pipeline`
- `metrics` Names of metrics to use, `NULL` for all (by default 8)
- `metrics_exclude` Names of metrics to exclude
- `n_models` Number of best models to return

Several approaches can be taken to sum the ranks of the models. To focus on the best models, we choose to rank only the top models for each metric and set all other to 0, instead of *e.g.* summing the ranks of all models over all metrics. This behavior is controlled by the `n_top` parameter

In a second step, we sum the ranks and return only the top models, and this is controlled by the `n_models` parameter. The output is a list of the rankings matrix and the selected models' parameters data frame (Table 1).

```
R> ensemble_metrics(n_top = 50, df_params = df_kxi)[[1]] %>%
+   print_vignette_table('Ensemble')
```

| avg.silwidth | bw.ratio | ch | pearsongamma | dunn | dunn2 | entropy | widestgap | sindex |
|--------------|----------|----|--------------|------|-------|---------|-----------|--------|
| 41 | 42 | 41 | 41 | 42 | 22 | 0 | 0 | 22 |
| 42 | 47 | 40 | 0 | 26 | 23 | 11 | 27 | 23 |
| 9 | 43 | 0 | 46 | 0 | 46 | 0 | 50 | 43 |
| 47 | 36 | 47 | 49 | 1 | 0 | 27 | 0 | 26 |
| 40 | 48 | 25 | 40 | 0 | 24 | 0 | 28 | 24 |
| 48 | 37 | 48 | 50 | 2 | 0 | 28 | 0 | 16 |
| 44 | 0 | 29 | 43 | 0 | 32 | 19 | 0 | 50 |
| 0 | 25 | 0 | 34 | 45 | 43 | 30 | 39 | 0 |
| 26 | 40 | 26 | 26 | 40 | 26 | 0 | 0 | 25 |
| 0 | 41 | 42 | 42 | 41 | 21 | 0 | 0 | 21 |

Table 1: In the first slot of the object returned by *ensemble_metrics*, we can investigate which metric voted for each model. Here we have the 10 models with highest sum of metrics ranks thresholded to 50. The top ensemble model was not the best model for any metric (otherwise the rank value would be 50), but was in the top 10 for 5 metrics (rank greater than 40), and was outside of the top 50 for 2 metrics (rank value set to 0).

Bootstrapping on several rank thresholds

The outer function is `ensemble_models` and is meant to be used on metrics objects computed with several different values of `n_top`. Above we have set `n_top = 50`, here we use 10%, 20%, and 50% of the number of models tested (Figure 5).

The first function returns 3 sets of top 10 models based on varying thresholds, and the second function chooses 4 models with most appearances in the top models. Both steps can produce ties, and in particular the second one. They are managed by ordering the models by increasing `n_xi`, `pts` (neighborhood size), and `n_dimred_comp` (number of dimension reduction components) to choose the least complex models possible.

```
R> df_ensemble_kxi = ensemble_models(df_kxi, n_models = 4,
+                                   model_subsample = c(0.1, 0.2, 0.5))
R> plot(gtable_kxi_profiles(df_ensemble_kxi))
```

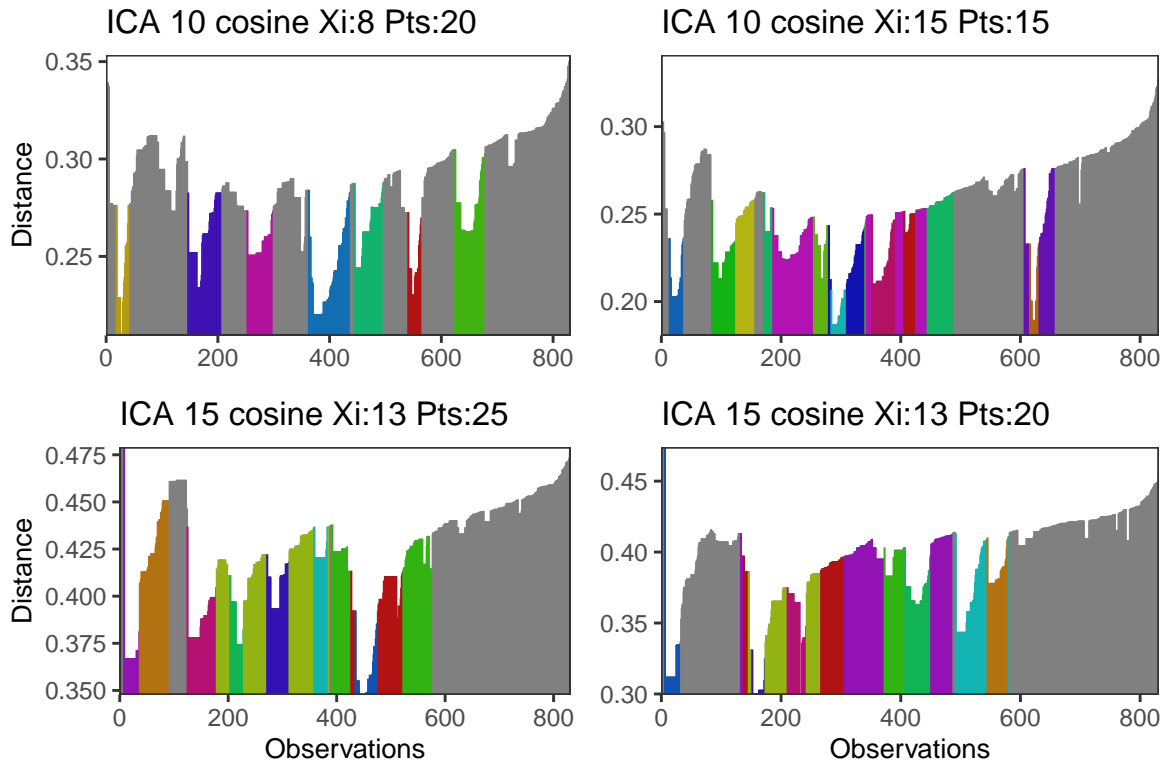


Figure 5: OPTICS k-Xi best 4 models for ensemble metrics, ordered by columns then rows. The best model corresponds to the second best Dunn index model.

3.3. Visualization with dimensionality reduction

We can visualize the clustering with dimensionality reduction. We performed the OPTICS k-Xi pipeline with independent component analysis (ICA) and the best model used 10 components. In contrast with principal component analysis (PCA), ICA does not order components and the results on less components will not be a subset of the components in higher dimensions. Still, for ease of visualization and summarization, here we display the clustering results obtained with 10 components on an ICA performed with only 4 components (Figure 6).

```
R> clusters = df_ensemble_kxi$clusters[[1]]
R> set.seed(1)
R> df_ica = fortify_ica(m_psych_embeds, n.comp = 4,
+                       sup_vars = data.frame(Clusters = clusters))
R> ggpairs(df_ica, 'Clusters', ellipses = TRUE, axes = 1:4) %>%
+   grid::grid.draw()
```

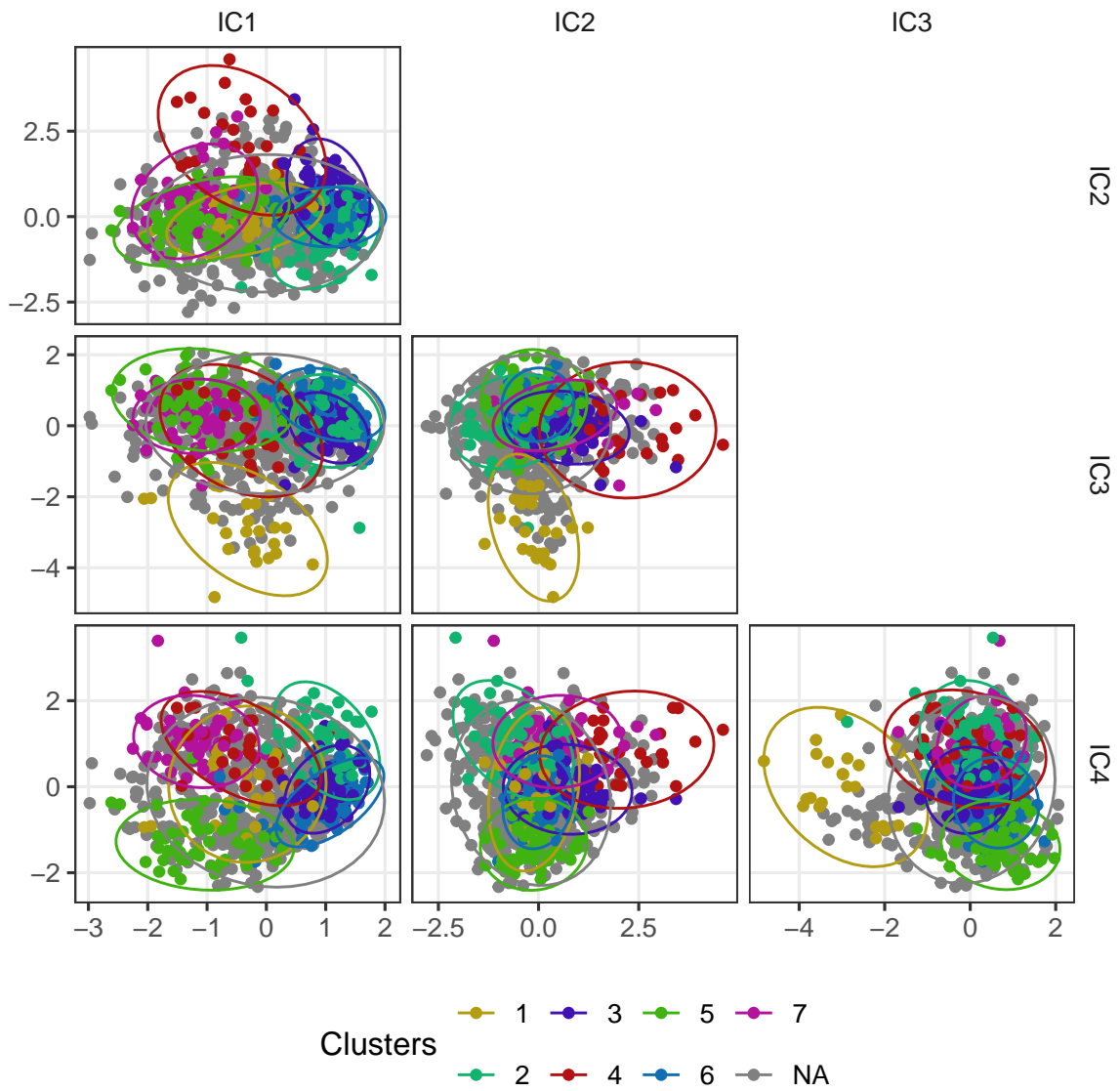


Figure 6: ICA dimensionality reduction, with ensemble clusters mapped.

For comparison, here are the clusters that would've been obtained using the model with best Dunn index metric (Figure 7).

```
R> best_kxi <- get_best_kxi(df_kxi, metric = 'dunn')
R> set.seed(1)
R> fortify_ica(m_psych_embeds, n.comp = 4,
+             sup_vars = data.frame(Clusters = best_kxi$clusters)) %>%
+   ggpairs('Clusters', ellipses = TRUE, axes = 1:4) %>%
+   grid::grid.draw()
```

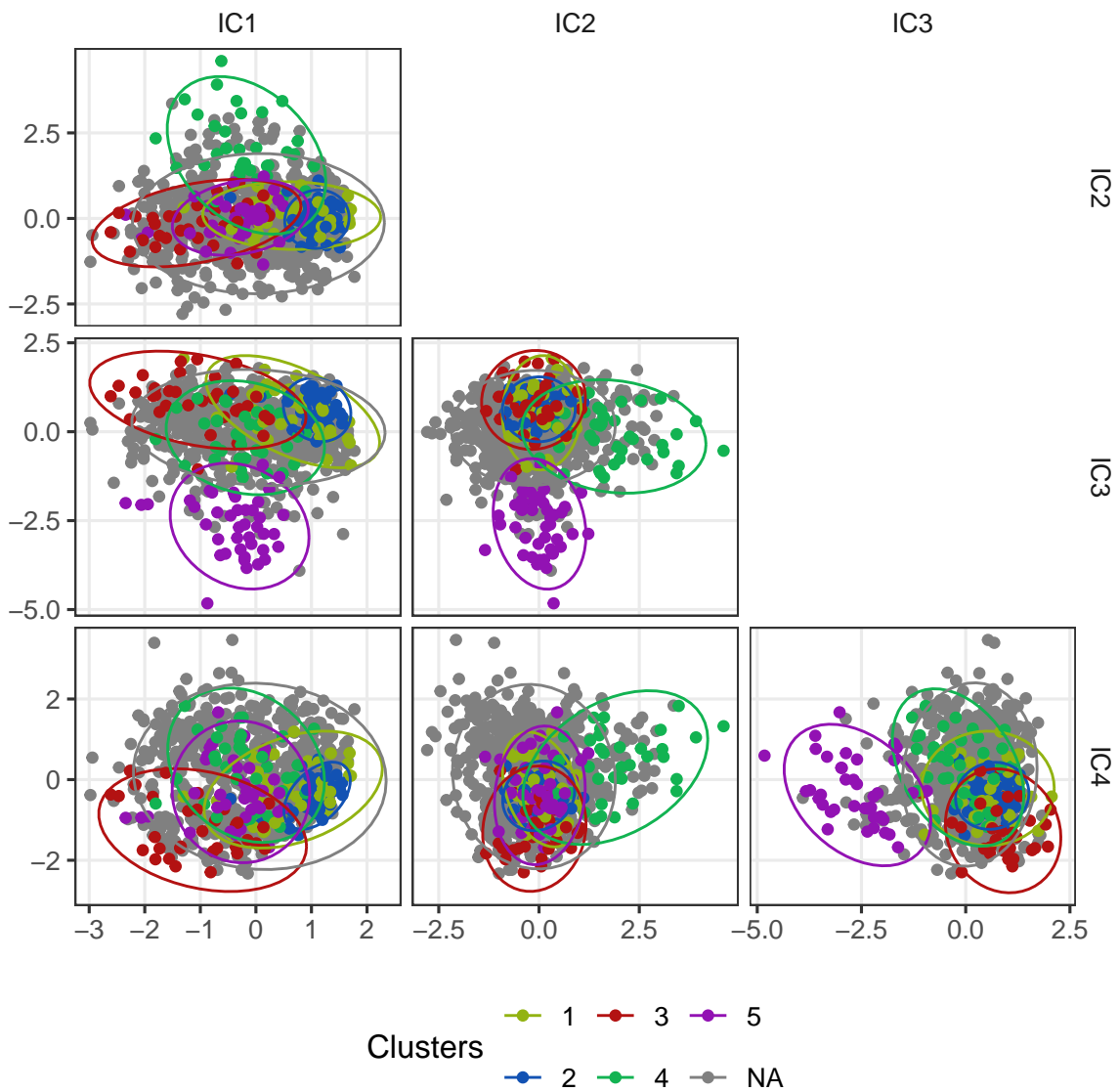


Figure 7: ICA dimensionality reduction, with best Dunn index clusters mapped.

4. Conclusions

This vignette showcased and demonstrated the use of ensemble metrics and models in the **opticskxi** package. While here the user could've manually investigated a few models and chosen the most appropriate one manually, these methods were implemented to enable chaining of several passes of OPTICS k-Xi which required an automated way of choosing the best clustering models. The study of word2vec embeddings on mental health open-access publications was presented at R in Medicine 2024 and the video is available online at <https://www.youtube.com/watch?v=sj7qYS8oRyc>.

References

Mikolov T (2013). "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781*, **3781**.

Affiliation:

Thomas Charlon
CELEHS Laboratory
Department of Biomedical Informatics
Harvard Medical School
10 Shattuck Street, Boston
E-mail: charlon@protonmail.com