Unsupervised Methods in the {tidymodels} Framework

Kelly Bodwin and Emil Hvitfeldt

2021-10-31

Signatories

Project team

The two PIs on this project are:

Dr. Kelly Bodwin: Kelly is an Assistant Professor of Statistics at California Polytechnic State University, San Luis Obispo. Her primary qualification for this project is expertise in unsupervised learning methods, which was the topic of her PhD dissertation as well as published papers in *Annals of Applied Statistics* and *Advances and Applications in Statistics*. Kelly also has extensive experience in teaching and using R and tidymodels, and recently published her first package on CRAN, `flair`.

Dr. Emil Hvitfeldt: Emil is a Data Analyst at Teladoc Health, a teaching adjunct professor at American University, and a co-author of the book Supervised Machine Learning for Text Analysis in R. He has built and maintains several R packages, including three packages in the `tidymodels` universe (`textrecipes`, `themis`, and `censored`). He brings to the project a high level of expertise in both the user-end and the developer side of `tidymodels`.

The team will provide roughly equal contribution to this project, with Emil leading the code design aspects of the project and Kelly leading the decisions about infrastructure and metrics for unsupervised settings.

Consulted

Both PIs have met individually in the past year with Max Kuhn and Julia Silge, who continue to express support and enthusiasm for this project. An intial planning conversation with the tidymodels team can be found on GitHub at this pull request.

The Problem

The advent of the `tidymodels` package suite in R marked a leap forward for making machine learning methods more accessible to a general audiences. However, this package suite is currently limited to the realm of supervised learning, in which observations of a target variable are used to train a predictive model. To implement *unsupervised* methods, such as clustering and community detection algorithms, users are still limited to standalone functions from outside the `tidymodels` framework. Users wishing to perform unsupervised analyses are unable to take advantage of the internal syntactical consistency and workflow-oriented structure of the `tidymodels` approach.

For example, two ubiquitous unsupervised algorithms are k-means clustering and hierarchical (agglomerative) clustering. In R, these are typically implemented using the `kmeans()` function and the `hclust()` functions, respectively, both from the base `stats` package. The functions differ greatly in format: `kmeans()` allows an input of a raw data table, `hclust()` **must** take a distance matrix; `kmeans()` produces cluster assignments while `hclust()` requires the additional use of `cutree()` or similar to label final clusters; and the two functions output different self-specific S3 object types.

As such, even if it is straightforward for an experienced R user to apply any one of these algorithms, each unique method requires starting from scratch in terms of data preparation and syntax. More importantly, enormous effort is needed to compare the results in any meaningful way, in order to select a final cluster assignment for the observations. These challenges mirror the pre-existing challenges in predictive modeling that were largely solved by `tidymodels` - making the remaing gap in available tools for unsupervised modeling all the more evident.

The proposal

Overview

We propose to develop two major R packages:

1. `celery`: This package will mirror the structure of the `parsnip` package in the `tidymodels` suite, with a focus on unsupervised methods.
2. `barometer`: This package will mirror the structure of the `yardstick` package in the `tidymodels` suite, implementing new and existing metrics that measure the quality of a clustering or community detection result.

These packages will be constructed to fit seamlessly alongside their supervised-learning cousins within the `tidymodels` structure; that is, they may be used with the resampling processes in `rsample`, the pre-processing structure of `workflows`, and the parameter tuning abilities of `tune` and `dials`.

Detail

Package `celery`

The first release of the `celery` package will provide a consistent syntax and framework for the following unsupervised methods:

- **k-means** and **k-medians**: Centroid-based method based on iterative maximization. Underlying implementation will be taken from the `stats::kmeans()` function.
- **Hierarchical clustering**: Agglomerative tree-based method. Underlying implementation from the `stats::hclust()` and `stats::cutree()` functions.
- **Model-based Clustering**: A density estimation method of Bayesian mixture models. Underlying implementation from the `mclust` package.
- **Network community detection**: Cluster extraction from graph structures. Underlying implementations taken from the `igraph` package.

It is important to note that we do not propose to implement any methods directly. Rather, we will establish an infrastructure and produce wrapper functions around existing code. For example, in the `parsnip` package, to specify a logistic regression model with penalization, the code is:

```
log_spec <- logistic_reg(penalty = 0.1) %>%
    set_mode("classification") %>%
    set_engine("glmnet")
```

In the `celery` package, the code to specify a k-means clustering might look like:

```
km_spec <- k_means(k = 5) %>%
    set_distance("euclidean") %>%
    set_mode("partition") %>%
    set_engine("kmeans")
```

The `celery` package will be released on CRAN, with through documentation including vignettes and references for each of the four major clustering method types that are implemented.

Package `barometer`

Results of unsupervised methods cannot be quantified in the same way as supervised models. In the supervised setting, one can compute residual prediction error by comparing the predicted target values to the true values, and compute common metrics like *r-squared* and *mean squared error* (for regression) and *precision*, *recall*, and *accuracy* (for classification). In the unsupervised setting, there is no target variable with known values, and so these common metrics do not apply.

The `barometer` package will develop, implement, and automate a set of metrics for cluster cohesion. Existing examples include:

- The *within sum of squares* to *between sum of squares* ratio, which measures relative cluster tightness.
- The *likelihood ratio*, which can compare two model-based clustering results.
- *Cluster consistency*, a measure of how similar clustering results are across different randomized initial conditions, different parameter choices, and/or different data subsamplings.

The `barometer` package will be released on CRAN, with through documentation including vignette walkthroughs of a method selection process relying on cross-validation over the implemented metrics.

Add-on functions

The integration of `celery` and `barometer` into the `tidymodels` framework will require a few standalone functions, that will either work with or be added to the existing `tidymodels` packages.

Some currently planned examples include:

- `step_cluster` function(s) for the `recipes` package. In some applications, unsupervised learning is used as a pre-processing step for supervised models.
- An unsupervised equivalent of `predict`, that applies the fitted unsupervised model to future data. This process is not as straightforward as `predict` - we sometimes want to assign new observations to the existing identified clusters, and we sometimes want to use the fitted parameters of the previous model to re-cluster a set of new observations.
- A set of `tidy` S3 methods for the `broom` package, that will reformat method output into clean data frames/tibbles.

Documentation: Use case and workflow

We note that for unsupervised learning, there are many contexts or use cases. Some use cases we have identified are:

- **Exploration:** The clustering or community detection process is applied to data with no prior goals or research questions, but simply as an exploratory step.
- **Validation:** Data comes with existing cluster membership labels, and the results of an unsupervised method are compared to these existing labels to determine how much the data supports the human-chosen groupings.
- **Pre-Processing:** Supervised models often call for *residualized* data, i.e., data where effects of superfluous qualities have been removed. An unsupervised method may be used to identify clusters in the data that are not intentional, or that distract from the research question; for example, one might identify patterns due to demographic information, like race or gender or age, that are not recorded in the data.

Each of these use cases lends itself to a slightly different workflow, and in particular, different metric needs in the `barometer` packages. In our final code release, we will explicitly identify the classes of use cases that `celery` is designed for, and provide thorough walkthroughs of the corresponding workflow for each case.

Project plan

Start-up phase

The planning phase is already well underway: the PIs meet regularly, and a detailed proposal is under discussion on GitHub. Upon funding confirmation, the PIs will make a public GitHub repository for this project, and begin contributing code immediately. Once the intial scaffolding is in place, the PIs will host an open forum advertised on twitter, to gather ideas and requests from the community at large.

Technical delivery

The project is anticipated to take one year.

The following checkpoints will keep the progress on target:

checkpoint	`celery` and `barometer`	documentation	other	community
Dec '21	GH Repository created with skeleton functions	aspirational workflow code	academic research conducted for unsupervised metrics	town hall held
Feb '21	alpha release of package	basic vignettes created	support functions shared with tidymodels team	invitation to test package
May '21	beta release of package	PR of support functions to tidymodels packages	use case specific vignettes created	application to summer conferences
July '21	submission to CRAN	full dedicated website with vignettes and walkthroughs		blog posts announcing package release
September '21	final CRAN revisions	video walkthroughs	discussion with tidymodels team for future plans	call for community input into future directions

Requirements

People

This project requires close and frequent contact with the `tidymodels` team, to ensure that our work is compatible with their current packages and their plans for future expansion.

Both PIs have met individually in the past year with Max Kuhn and Julia Silge, who continue to express support and enthusiasm for this project. An intial planning conversation with the tidymodels team can be found on GitHub at this pull request.

We fully expect this line of communication to remain open as we develop the software. We will also seek input from the broader R community, via twitter, RStudio forums, etc.

Processes

We will employ the code of conduct developed by rOpenSci, which can be found at https://ropensci.org/code-of-conduct/.

Tools & Tech

All technical tools and technologies to deliver the outlined developments are already in place.

Funding

We estimate a combined effort of 200 hours will be necessary to complete the development of this software. We are requesting $12,000 to compensate this time.

Compensation was calculated based on the smaller of the two hourly billing rates of the two PIs.

Success

Definition of done

This project is complete when the following conditions are met:

1. `celery` is released on CRAN, with at least four methods (k-means, hierarchical clustering, model-based clustering, network community detection) included.
2. `barometer` is released on CRAN, with the ability to compute multiple metrics for all four `celery` methods, and to work seamlessly with `rsample` to cross-validate metrics.
3. Necessary additional functions are written and pull-requested to their corresponding `tidymodels` package.
4. Documentation is complete for multiple use cases of all methods, including vignettes that demonstrate a `tidymodels`-integrated workflow.
5. PIs have applied to present results at least one of the following conferences:
   RStudio::conf, UseR!, Joint Statistical Meetings, ASA Symposium on Data Science and Statistics.

Measuring success

The creation of vignettes will also provide a measure of success; if the `tidymodels` workflow cannot be demonstrated thoroughly, then `celery` and `barometer` are not yet complete. We will also rely on community feedback, via intermediate blog posts and twitter calls for input, to measure the usability and clarity of our code design.

Future work

There is an enormous possibility for expansion via inclusion of further unsupervised methods in `celery`. We anticipate that community members will frequently request, or implement themselves, other algorithms of interest. It is likely that such expansion will uncover further nuance in the general principles of unsupervised learning, and we will need to add to the underlying `celery` infrastructure to meet these new challenges. Similarly, we also anticipate that additional `barometer` metrics will be suggested or contributed by the community after release.

Key risks

The primary risk lies in compatibility with existing `tidymodels` framework. It is possible that the design of an unsupervised learning infrastructure necessitates changes to certain elements of `tidymodels`. This would require a level of contribution and collaboration from the tidymodels team beyond the general support and advising they have currently committed. We are confident that our currently planned designs can intregrate as expected; should new complexity emerge in the process, we will revisit plans with the tidymodels team, and possibly build our own scaffolding rather than rely on existing packages.

Acknowledgement

Many of the ideas influencing the planning of this project are taken from the paper Clustering: Science or Art? (Luxberg, Williamson, Guyon; 2012).

This proposal is based on the boilerplate by Ben Graeler.