Package: polarisR (via r-universe)

May 17, 2026

Title Non-Linear Dimensionality Reduction Visualization Tool

Version 0.1.5

Description A 'shiny' application for visualizing high-dimensional data using non-linear dimensionality reduction (NLDR) techniques such as t-SNE and UMAP. It provides an interactive platform to explore high-dimensional datasets, diagnose the quality of the embeddings using the 'quollr' package, and compare different NLDR methods.

License MIT + file LICENSE

URL https://github.com/Divendra2006/polarisR, https://divendra2006.github.io/polarisR/

BugReports https://github.com/Divendra2006/polarisR/issues

Depends R (>= 4.1.0)

Imports bslib, crosstalk, detourr, dplyr, DT, FNN, future, ggplot2, magrittr, parallelly, plotly, quollr, Rtsne, scales, shiny, stats, tools, tourr, umap, utils

Suggests knitr, rmarkdown, spelling, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs cmake libglpk-dev make libgsl0-dev libicu-dev libpng-dev libuv1-dev libxml2-dev libssl-dev python3 zlib1g-dev

Repository https://divendra2006.r-universe.dev

Date/Publication 2026-04-17 10:16:51 UTC

RemoteUrl https://github.com/divendra2006/polarisr

RemoteRef HEAD

RemoteSha 56eb811fc3bdb63c4cfa6931886178728d2a0173

Contents

`four_clusters` *Four Clusters Simulated Dataset*

Description

This dataset (`four_clusters`) contains simulated data with four distinct clusters, each generated using different shapes and scales. It is ideal for demonstrating clustering algorithms and visualization techniques.

Usage

```
four_clusters
```

Format

A data frame with 2000 rows and 4 variables:

x1 Numeric. First feature coordinate.

x2 Numeric. Second feature coordinate.

x3 Numeric. Third feature coordinate.

x4 Numeric. Fourth feature coordinate.

cluster Factor. The cluster label (1, 2, 3, or 4).

Details

Four Clusters Simulated Dataset

Source

Simulated using `cardinalR` package.

References

Gamage J, Cook D, Harrison P, Lydeamore M, Talagala T (2025).*cardinalR: Collection of Data Structures*. R package version 0.1.10, https://github.com/JayaniLakshika/cardinalR.

Examples

```
data(four_clusters)
head(four_clusters)
dim(four_clusters)
```

`load_custom_datasets` *Load Built-in Datasets for NLDR Visualization*

Description

Loads and returns a named list of built-in example datasets that are included with the polarisR package. These datasets are specifically chosen to demonstrate different characteristics and challenges in non-linear dimensionality reduction analysis and visualization.

Usage

```
load_custom_datasets()
```

Value

A named list containing the successfully loaded datasets. Each element is a data.frame with the following structure:

- `$four_clusters`: data.frame with cluster analysis data
- `$pdfsense`: data.frame with high-dimensional physics data

If any dataset fails to load, a warning is issued but the function continues, returning only the successfully loaded datasets. An empty list is returned if no datasets can be loaded.

Error Handling

The function includes robust error handling:

- Each dataset is loaded in a separate `tryCatch` block
- Loading failures generate warnings rather than stopping execution
- Existence checks ensure datasets are properly loaded before inclusion
- Environment management prevents namespace pollution

Usage in Application

This function is typically called during application initialization to populate the example dataset dropdown in the UI. The returned list is stored in a reactive value and can be dynamically extended with user-uploaded datasets.

Note

The function uses `data` to load datasets from the package namespace. If the polarisR package is not properly installed or the data files are missing, warnings will be generated for the affected datasets.

Author(s)

GSoC Contributor

See Also

- four_clusters for four_clusters dataset documentation
- pdfsense for pdfsense dataset documentation
- data for dataset loading mechanism

Examples

```
if(interactive()){
# Load all available datasets
datasets <- load_custom_datasets()
names(datasets)  # Shows available dataset names

# Check what was successfully loaded
if ("four_clusters" %in% names(datasets)) {
  dim(datasets$four_clusters)
  head(datasets$four_clusters)
}

# Use in shiny application initialization
custom_datasets <- shiny::reactiveVal(load_custom_datasets())
available_datasets <- shiny::reactiveVal(
  c("None", names(load_custom_datasets()))
)
}
```

---

nldr_viz_server *NLDR Visualization Tool Server Logic*

Description

Implements the comprehensive server-side logic for the NLDR visualization tool. This function handles all reactive computations, data processing, visualization generation, and user interactions for the multi-tab NLDR analysis application.

Usage

```
nldr_viz_server(input, output, session)
```

Arguments

`input`	The shiny input object containing all user interface inputs. This includes form controls, button clicks, plot selections, and file uploads.
`output`	The shiny output object for sending rendered content to the UI. Used for plots, tables, text outputs, and dynamic UI elements.
`session`	The shiny session object for managing client-server communication. Provides access to session state, input updates, and client information.

Details

The server function manages several key areas of functionality:

Data Management:

- File upload validation and CSV parsing with error handling
- Built-in dataset loading (four_clusters, pdfsense)
- Empty cell detection and data quality validation
- Column selection and filtering capabilities
- Dynamic dataset storage and retrieval system

NLDR Computations:

- t-SNE implementation with parameter validation and auto-adjustment
- UMAP processing with neighbor and distance parameter controls
- Asynchronous computation using future package for responsiveness
- Progress tracking and user feedback during long computations
- Result caching and session management

Interactive Visualizations:

- Plotly-based interactive scatter plots with zoom, pan, and selection
- Color mapping with automatic palette generation
- Linked brushing across multiple visualizations
- Responsive plot sizing and layout management
- Hover tooltips and selection feedback

Dynamic Tours:

- Integration with detourr package for animated projections
- Multiple display types: Scatter, Sage, and Slice projections
- 5-nearest neighbor graph construction and visualization
- Real-time parameter adjustment and tour customization
- Coordinated views between static and dynamic visualizations

Quality Assessment (Quollr Integration):

- Automated binwidth optimization using RMSE minimization
- Hexagonal binning and centroid extraction
- High-dimensional model fitting and validation
- Prediction error analysis and visualization
- 3D model tours using langevitour integration

Method Comparison:

- Side-by-side visualization comparison with linked brushing
- RMSE-based parameter optimization comparison
- Best configuration identification and reporting
- Interactive comparison plot generation

State Management:

- Reactive value system for maintaining application state
- Session-based data persistence and cleanup
- Asynchronous operation tracking and user feedback
- Memory management for large datasets

Parallel Processing: The server automatically configures parallel processing:

- Uses multicore on supported systems, multisession otherwise
- Configures 2 worker processes for optimal performance
- Proper cleanup on session end to prevent memory leaks
- Future-based asynchronous computation for UI responsiveness

Value

Invisible NULL. The function sets up reactive expressions and observers that handle all server-side logic. The actual outputs are managed through the shiny reactive system and sent to the client via the output object.

Reactive Values

The server maintains several key reactive values:

- `dataset`: Current active dataset
- `vis_results`: NLDR computation results
- `shared_vis_data`: Crosstalk-enabled data for linked brushing
- `nldr_datasets`: Storage for multiple NLDR results
- `optimal_config`: Best binwidth configuration from optimization
- `quollr_results`: Quality assessment results
- `color_palette`: Current color scheme for visualizations
- `is_running_*`: Boolean flags for operation status tracking

Error Handling

Comprehensive error handling includes:

- File upload validation with user-friendly error messages
- NLDR computation error catching with fallback options
- Memory management for large datasets
- Network timeout handling for async operations
- Graceful degradation when optional features are unavailable

Performance Considerations

- Asynchronous computations prevent UI blocking
- Efficient data structures for large datasets
- Caching of expensive computations
- Memory cleanup and garbage collection
- Optimized reactive dependency management

Note

This function requires several packages to be available:

- **Core**: shiny, magrittr
- **Visualization**: plotly, ggplot2, DT, scales
- **NLDR**: Rtsne, umap, FNN
- **Quality**: quollr (with all its dependencies)
- **Tours**: detourr, tourr
- **Data**: dplyr, crosstalk
- **Async**: future
- **Utils**: stats, utils, tools

Author(s)

GSoC Contributor

See Also

- `nldr_viz_ui` for the corresponding user interface
- `run_nldr_viz` for launching the complete application
- `load_custom_datasets` for data loading utilities
- `shinyServer` for shiny server function details
- `plan` for parallel processing configuration

`nldr_viz_ui` *NLDR Visualization Tool User Interface*

Description

Creates the comprehensive user interface for the NLDR visualization tool using modern Bootstrap 5 styling with bslib. The UI provides an intuitive tabbed interface for data exploration, dimensionality reduction, quality assessment, and method comparison.

Usage

```
nldr_viz_ui()
```

Details

The user interface is organized into five main navigation panels:

1. Dataset Preview Tab:

- File upload widget for CSV datasets with validation
- Selection of built-in example datasets (four_clusters, pdfsense)
- Interactive column selection with auto-selection option
- Data preview table with scrolling and pagination
- Dataset summary statistics (rows, columns, data types)
- Storage and management of processed NLDR results

2. Dataset Visualization Tab:

- Method selection: t-SNE or UMAP with parameter controls
- t-SNE parameters: perplexity (5-50), max iterations (100-2000), auto-adjust option
- UMAP parameters: n_neighbors (2-50), min_dist (0.01-0.99)
- Color mapping options with automatic or manual column selection
- Reproducibility controls with random seed setting
- Large interactive visualization plot (800px height)
- Visualization information panel with method details

3. Dynamic Tour Tab:

- Tour display types: Scatter, Sage, or Slice projections
- Display customization: axes, 5-NN graph edges, point opacity
- Type-specific parameters (alpha for Scatter, gamma for Sage, volume for Slice)
- Linked brushing functionality for interactive selection
- Side-by-side layout: NLDR plot and dynamic tour visualization

4. Diagnosing Tab:

- Automated binwidth optimization with RMSE-based selection
- Comprehensive quollr analysis for embedding quality assessment
- 3D model tour generation with langevitour integration
- Results presentation in multiple tabs:
  - RMSE vs Binwidth interactive plot
  - Optimization results table with sortable columns
  - Model fit visualization showing prediction quality
  - High-dimensional model tour for deeper exploration
- Configuration summary with optimal parameter display

5. Method Comparison Tab:

- Two comparison modes: NLDR Settings Comparison and Side-by-Side Visualization
- Settings Comparison: RMSE comparison plots across multiple configurations
- Side-by-Side: Interactive linked plots for direct method comparison
- Dataset selection controls with stored results integration
- Linked brushing option for synchronized selections
- Best configuration summary and recommendations

The interface uses modern design principles:

- Responsive layout that adapts to different screen sizes
- Consistent spacing and typography using Bootstrap 5
- Loading indicators and progress feedback for long-running operations
- Intuitive iconography and color coding for different actions
- Collapsible sidebar layouts for optimal screen real estate usage
- Contextual help text and tooltips for user guidance

Value

A `tagList` object representing the complete UI structure suitable for use in `shinyApp` or as a module UI function. The returned object contains all necessary HTML, CSS, and JavaScript dependencies.

Custom CSS

The interface includes custom CSS for:

- Disabled button states with reduced opacity
- Spinning loading animations for progress indication
- Consistent button styling across different states
- Responsive card layouts and spacing

Accessibility

The UI follows accessibility best practices:

- Proper ARIA labels and roles for interactive elements
- Keyboard navigation support throughout the interface
- High contrast color scheme for readability
- Screen reader compatible structure and labeling

Note

The UI function automatically imports all required dependencies:

- `bslib` for modern Bootstrap 5 theming (Lumen theme)
- `plotly` for interactive visualizations
- `DT` for enhanced data tables
- Custom CSS for loading animations and button states
- Font Awesome icons for consistent iconography

Author(s)

GSoC Contributor

See Also

- `nldr_viz_server` for corresponding server logic
- `run_nldr_viz` for launching the complete application
- `page_navbar` for navigation structure
- `plotlyOutput` for interactive plot outputs
- `DTOutput` for enhanced data table displays

`pdfsense` *PDFSense Dataset*

Description

A dataset representing the space of parton distribution function fit, each point in the variables labelled X1-X56 indicate moving +- 1 standard devation from the 'best' (maximum likelihood estimate) fit of the function. Each observation has all predictions of the corresponding measurement from an experiment.

Usage

```
pdfsense
```

Format

An object of class `data.frame` with 2808 rows and 62 columns.

Details

(see table 3 in that paper for more explicit details).

The remaining columns are:

- InFit: A flag indicating whether an observation entered the fit of CT14HERA2 parton distribution function
- Type: First number of ID
- ID: contains the identifier of experiment, 1XX/2XX/5XX correpsonds to Deep Inelastic Scattering (DIS) / Vector Boson Production (VBP) / Strong Interaction (JET). Every ID points to an experimental paper.
- pt: the per experiment observational id
- x,mu: the kinematics of a parton. x is the parton momentum fraction, and mu is the factorisation scale.

Source

http://www.physics.smu.edu/botingw/PDFsense_web_histlogy/

References

Wang, B.-T., Hobbs, T. J., Doyle, S., Gao, J., Hou, T.-J., Nadolsky, P. M., & Olness, F. I. (2018). PDFSense: Mapping the sensitivity of hadronic experiments to nucleon structure. Retrieved from https://arxiv.org/abs/1808.07470

Cook, D., Laa, U., & Valencia, G. (2018). Dynamical projections for the visualization of PDFSense data. The European Physical Journal C, 78(9), 742. doi:10.1140/epjc/s1005201862052

`run_nldr_viz` *Run NLDR Visualization Tool*

Description

Launches the interactive NLDR (Non-Linear Dimensionality Reduction) Visualization Tool as a shiny application for exploring high-dimensional datasets using t-SNE and UMAP.

Usage

```
run_nldr_viz(...)
```

Arguments

`...`	Additional arguments passed to `shinyApp`

Details

The application includes five main tabs:

- **Dataset Preview**: Upload data and preview dataset characteristics
- **Non-linear dimension reduction (NLDR)**: Apply t-SNE or UMAP methods
- **Dynamic Tour**: Explore high-dimensional structure through animated projections
- **Diagnosing**: Assess embedding quality using quollr package
- **2-D Layout Comparison**: Compare different NLDR configurations

Value

A shiny application object that launches the NLDR visualization interface.

See Also

`nldr_viz_ui`, `nldr_viz_server`

Index