Package ‘bettr’

May 29, 2024

Title  A Better Way To Explore What Is Best
Version  1.0.0
Date  2024-03-24
Description  bettr provides a set of interactive visualization methods to explore the results of a benchmarking study, where typically more than a single performance measures are computed. The user can weight the performance measures according to their preferences. Performance measures can also be grouped and aggregated according to additional annotations.
License  MIT + file LICENSE
Encoding  UTF-8
Suggests  knitr, rmarkdown, testthat (>= 3.0.0), BiocStyle
VignetteBuilder  knitr
RoxygenNote  7.3.1
Roxygen  list(markdown = TRUE)
Depends  R (>= 4.4.0)
Imports  dplyr (>= 1.0), tidyr, ggplot2 (>= 3.4.1), shiny (>= 1.6), tibble, ComplexHeatmap, bslib, rlang, circlize, stats, grid, methods, cowplot, Hmisc, sortable, shinyjqui, grDevices, scales, DT, SummarizedExperiment, S4Vectors
Config/testthat/edition  3
biocViews  Visualization, ShinyApps, GUI
URL  https://github.com/federicomarini/bettr
BugReports  https://github.com/federicomarini/bettr/issues
git_url  https://git.bioconductor.org/packages/bettr
git_branch  RELEASE_3_19
git_last_commit  fa4e76a
git_last_commit_date  2024-04-30
Repository  Bioconductor 3.19
Date/Publication  2024-05-29
bettr-package

Author Federico Marini [aut] (<https://orcid.org/0000-0003-3252-7758>),
Charlotte Soneson [aut, cre] (<https://orcid.org/0000-0003-3833-2169>)

Maintainer Charlotte Soneson <charlottesoneson@gmail.com>

Contents

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>bettr-package</td>
<td>2</td>
</tr>
<tr>
<td>assembleSE</td>
<td>3</td>
</tr>
<tr>
<td>bettr</td>
<td>5</td>
</tr>
<tr>
<td>bettrGetReady</td>
<td>7</td>
</tr>
<tr>
<td>makeBarPolarPlot</td>
<td>10</td>
</tr>
<tr>
<td>makeHeatmap</td>
<td>12</td>
</tr>
<tr>
<td>makeParCoordPlot</td>
<td>14</td>
</tr>
<tr>
<td>makePolarPlot</td>
<td>16</td>
</tr>
</tbody>
</table>

Index 18

bettr-package bettr: a better way to explore what is best

Description

The bettr package provides a better way to explore what is best :) Details about how to use the package can be found in the vignette. The main entry point is the `bettr()` function, which opens an interactive application for exploring data consisting of multiple parallel rankings of a set of entities (e.g., computational methods ranked by their performance based on several different metrics).

Author(s)

Charlotte Soneson <charlottesoneson@gmail.com>

Federico Marini <marinif@uni-mainz.de>

See Also

Useful links:

- [https://github.com/federicomarini/bettr](https://github.com/federicomarini/bettr)
- Report bugs at [https://github.com/federicomarini/bettr/issues](https://github.com/federicomarini/bettr/issues)
assembleSE

Assemble all bettr input into a SummarizedExperiment object

Description

Assemble all bettr input into a SummarizedExperiment object. This has the advantage of keeping all data together in a single object, and can be used as input to bettr or bettrGetReady, instead of providing the individual components.

Usage

```r
assembleSE(
  df,
  idCol = "Method",
  metrics = setdiff(colnames(df), idCol),
  initialWeights = NULL,
  initialTransforms = list(),
  metricInfo = NULL,
  metricColors = NULL,
  idInfo = NULL,
  idColors = NULL
)
```

Arguments

df  A data.frame in wide format. Should contain one column with the IDs of the entities to be compared, and one column for each metric to use for the comparison.

idCol  Character scalar, indicating the name of the column of df and/or idInfo that contains IDs of the entities to be compared (e.g., methods).

metrics  Character vector, indicating which of the columns of df that correspond to metrics of interest. Only metrics included here will be displayed.

initialWeights  Named numeric vector providing initial weights for each metric to use for aggregating them into a final score. Must contain one entry per metric included in metrics.

initialTransforms  Named list with initial values of transformation parameters for each metric. Each list entry should correspond to one metric, and take the form of a list with up to four elements, named:

* **flip**: Logical scalar; whether or not to flip the sign of the metric values. Defaults to 'FALSE'.
* **offset**: Numeric scalar; offset to add to the (flipped) metric values. Defaults to '0'.
* **transform**: Character scalar; one of 'None', 'z-score', '\[0,1\]', '\[-1,1\]', 'Rank', 'Rank+\[0,1\]' or 'z-score+\[0,1\]'.
indicating which transform to apply to the metric values (after any flipping and/or adding the offset). Defaults to 'None'.

* **cuts**: Numeric vector or `NULL`; the cut points that will be used to bin the metric values (after the other transformations). Defaults to `NULL`.

Only values deviating from the defaults need to be explicitly specified, the others will be initialized to their default values.

metricInfo  
data.frame with annotations for metrics. Must have a column named 'Metric' identifying the respective metrics.

metricColors  
Named list with colors used for columns of metricInfo. Should follow the format required for ComplexHeatmap heatmap annotations. The list can include an entry named 'Metric', which contains a named vector with colors to use for metrics.

idInfo  
data.frame with annotations for entities. Must have a column named according to idCol identifying the respective entities.

idColors  
Named list with colors used for columns of idInfo. Should follow the format required for ComplexHeatmap heatmap annotations. The list can include an entry named according to idCol, which contains a named vector with colors to use for entities.

Value

A SummarizedExperiment object with rows corresponding to methods and columns corresponding to metrics.

Author(s)

Charlotte Soneson

Examples

df <- data.frame(Method = c("M1", "M2", "M3"),
  metric1 = c(1, 2, 3),
  metric2 = c(3, 1, 2))
metricInfo <- data.frame(Metric = c("metric1", "metric2", "metric3"),
  Group = c("G1", "G2", "G2"))
idInfo <- data.frame(Method = c("M1", "M2", "M3"),
  Type = c("T1", "T1", "T2"))
bettrSE <- assembleSE(df = df, metricInfo = metricInfo, idInfo = idInfo)
Launch bettr app to explore and aggregate performance metrics

**Description**
Launch bettr app to explore and aggregate performance metrics

**Usage**
```r
bettr(
  df,
  idCol = "Method",
  metrics = setdiff(colnames(df), idCol),
  initialWeights = NULL,
  initialTransforms = list(),
  metricInfo = NULL,
  metricColors = NULL,
  idInfo = NULL,
  idColors = NULL,
  weightResolution = 0.05,
  bstheme = "darkly",
  appTitle = "bettr",
  bettrSE = NULL
)
```

**Arguments**
- **df**
  A data.frame in wide format. Should contain one column with the IDs of the entities to be compared, and one column for each metric to use for the comparison.
- **idCol**
  Character scalar, indicating the name of the column of df and/or idInfo that contains IDs of the entities to be compared (e.g., methods).
- **metrics**
  Character vector, indicating which of the columns of df that correspond to metrics of interest. Only metrics included here will be displayed.
- **initialWeights**
  Named numeric vector providing initial weights for each metric to use for aggregating them into a final score. Must contain one entry per metric included in metrics.
- **initialTransforms**
  Named list with initial values of transformation parameters for each metric. Each list entry should correspond to one metric, and take the form of a list with up to four elements, named:
  - **flip**
    Logical scalar; whether or not to flip the sign of the metric values. Defaults to `FALSE`.
  - **offset**
    Numeric scalar; offset to add to the (flipped) metric values. Defaults to `0`.  

bettr

* **transform**: Character scalar; one of 'None', 'z-score', '[0,1]', '[-1,1]', 'Rank', 'Rank+[0,1]' or 'z-score+[0,1]', indicating which transform to apply to the metric values (after any flipping and/or adding the offset). Defaults to 'None'.

* **cuts**: Numeric vector or 'NULL'; the cut points that will be used to bin the metric values (after the other transformations). Defaults to 'NULL'.

Only values deviating from the defaults need to be explicitly specified, the others will be initialized to their default values.

metricInfo data.frame with annotations for metrics. Must have a column named 'Metric' identifying the respective metrics.

metricColors Named list with colors used for columns of metricInfo. Should follow the format required for ComplexHeatmap heatmap annotations. The list can include an entry named 'Metric', which contains a named vector with colors to use for metrics.

idInfo data.frame with annotations for entities. Must have a column named according to idCol identifying the respective entities.

idColors Named list with colors used for columns of idInfo. Should follow the format required for ComplexHeatmap heatmap annotations. The list can include an entry named according to idCol, which contains a named vector with colors to use for entities.

weightResolution Numeric scalar in (0,1), giving the resolution at which weights can be specified using the sliders in the interface.

bstheme Character scalar giving the bootswatch theme for the app (see https://bootswatch.com/). Default 'darkly'.

appTitle Character scalar giving the title that will be used for the app. Defaults to 'bettr'.

bettrSE A SummarizedExperiment generated by assembleSE. If this is not NULL, df, metrics, initialWeights, initialTransforms, metricInfo, metricColors, idInfo and idColors arguments will be ignored and the information will be extracted from the SummarizedExperiment object.

Value
A shiny application

Author(s)
Charlotte Soneson

Examples
df <- data.frame(Method = c("M1", "M2", "M3"), metric1 = c(1, 2, 3), metric2 = c(3, 1, 2), metric3 = factor(c("a", "a", "b")))
initialTransforms <- list(metric1 = list(flip = TRUE, offset = 4))
metricInfo <- data.frame(Metric = c("metric1", "metric2", "metric3"),
                         Group = c("G1", "G2", "G2"))
idInfo <- data.frame(Method = c("M1", "M2", "M3"),
                     Type = c("T1", "T1", "T2"))
metricColors <- list(Group = c(G1 = "red", G2 = "blue"))
if (interactive()) {
  bettr(df = df, idCol = "Method",
       metrics = c("metric1", "metric2", "metric3"),
       initialTransforms = initialTransforms,
       metricInfo = metricInfo, metricColors = metricColors,
       idInfo = idInfo)
}

---

bettrGetReady  Prepare data for plotting with bettr

Description

Prepare input data for plotting with bettr. This function replicates the steps that are performed in the shiny app.

Usage

bettrGetReady(
  df,
  idCol = "Method",
  metrics = setdiff(colnames(df), idCol),
  initialWeights = NULL,
  initialTransforms = list(),
  metricInfo = NULL,
  metricColors = NULL,
  idInfo = NULL,
  idColors = NULL,
  scoreMethod = "weighted mean",
  idOrdering = "high-to-low",
  showOnlyTopIds = FALSE,
  nbrTopIds = 10,
  idTopNGrouping = NULL,
  keepIds = NULL,
  metricGrouping = NULL,
  metricCollapseGroup = FALSE,
  metricCollapseMethod = "mean",
  bettrSE = NULL
)
Arguments

df A data.frame in wide format. Should contain one column with the IDs of the entities to be compared, and one column for each metric to use for the comparison.

idCol Character scalar, indicating the name of the column of df and/or idInfo that contains IDs of the entities to be compared (e.g., methods).

metrics Character vector, indicating which of the columns of df that correspond to metrics of interest. Only metrics included here will be displayed.

initialWeights Named numeric vector providing initial weights for each metric to use for aggregating them into a final score. Must contain one entry per metric included in metrics.

initialTransforms Named list with initial values of transformation parameters for each metric. Each list entry should correspond to one metric, and take the form of a list with up to four elements, named:

* **flip**: Logical scalar; whether or not to flip the sign of the metric values. Defaults to 'FALSE'.
* **offset**: Numeric scalar; offset to add to the (flipped) metric values. Defaults to '0'.
* **transform**: Character scalar; one of 'None', 'z-score', '\[0,1\]', '\[-1,1\]', 'Rank', 'Rank+\[0,1\]' or 'z-score+\[0,1\]', indicating which transform to apply to the metric values (after any flipping and/or adding the offset). Defaults to 'None'.
* **cuts**: Numeric vector or 'NULL'; the cut points that will be used to bin the metric values (after the other transformations). Defaults to 'NULL'.

Only values deviating from the defaults need to be explicitly specified, the others will be initialized to their default values.

metricInfo data.frame with annotations for metrics. Must have a column named 'Metric' identifying the respective metrics.

metricColors Named list with colors used for columns of metricInfo. Should follow the format required for ComplexHeatmap heatmap annotations. The list can include an entry named 'Metric', which contains a named vector with colors to use for metrics.

idInfo data.frame with annotations for entities. Must have a column named according to idCol identifying the respective entities.

idColors Named list with colors used for columns of idInfo. Should follow the format required for ComplexHeatmap heatmap annotations. The list can include an entry named according to idCol, which contains a named vector with colors to use for entities.

scoreMethod Character scalar specifying the scoring method, that is, how to aggregate scores across metrics. Should be one of "weighted mean", "weighted median", "weighted fraction highest" or "weighted fraction lowest".


idOrdering Character scalar indicating whether methods should be ranked with highest aggregated scores on top ("high-to-low") or opposite ("low-to-high").

showOnlyTopIds Logical scalar indicating whether to only retain the top N methods (ranked by the aggregated score).

nbrTopIds If showOnlyTopIds is TRUE, the number of top-ranked methods to retain.

idTopNGrouping If showOnlyTopIds is TRUE, a character scalar providing the name of a column in idInfo that groups the methods. If specified, the top nbrTopIds within each group will be retained.

keepIds Character vector indicating which methods (a subset of the values in df[[idCol]]) that should be considered. If NULL, all methods are considered.

metricGrouping A character scalar providing the name of a column in metricInfo by which metrics should be grouped. If NULL, no grouping is performed.

metricCollapseGroup A logical scalar indicating whether metric values should be collapsed within each group defined by metricGrouping.

metricCollapseMethod If metricCollapseGroup is TRUE, the way in which metric values are collapsed within a group. Should be one of "mean", "max" or "min".

bettrSE A SummarizedExperiment generated by assembleSE. If this is not NULL, df, metrics, initialWeights, initialTransforms, metricInfo, metricColors, idInfo and idColors arguments will be ignored and the information will be extracted from the SummarizedExperiment object.

Value

A list of objects, which can be directly used as inputs for the bettr plotting functions. See the man page for the respective plotting function for more details.

Author(s)

Charlotte Soneson

Examples

## Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),
               metric1 = c(1, 2, 3),
               metric2 = c(3, 1, 2))
metricInfo <- data.frame(Metric = c("metric1", "metric2", "metric3"),
                         Group = c("G1", "G2", "G2"))
idInfo <- data.frame(Method = c("M1", "M2", "M3"),
                     Type = c("T1", "T1", "T2"))
prepData <- bettrGetReady(df = df, idCol = "Method",
                           metricInfo = metricInfo, idInfo = idInfo)
prepData <- bettrGetReady(df = df, idCol = "Method",
                           metricInfo = metricInfo, idInfo = idInfo,
                           metricGrouping = "Group",
                           metricCollapseGroup = TRUE)
makeBarPolarPlot

Create a bar/polar plot

Description

Create a bar/polar plot. The input arguments for this function are typically generated using `bettrGetReady`, which ensures that all required columns are available.

Usage

```r
makeBarPolarPlot(
  bettrList = NULL,
  plotdata,
  scoredata,
  idCol,
  metricCol = "Metric",
  valueCol = "ScaledValue",
  weightCol = "Weight",
  scoreCol = "Score",
  metricGroupCol = "metricGroup",
  metricColors,
  metricCollapseGroup = FALSE,
  metricGrouping = "---",
  methods = NULL,
  labelSize = 10,
  showComposition = FALSE,
  scaleFactorPolars = 1
)
```

Arguments

- **bettrList**: A list, the output object from `prepData`. If `bettrList` is provided, arguments `plotdata`, `scoredata`, `idCol`, `metricCol`, `valueCol`, `weightCol`, `scoreCol`, `metricGroupCol`, `metricInfo`, `metricColors`, `idInfo`, `idColors`, `metricCollapseGroup`, `metricGrouping` and `methods` will be ignored and the corresponding values will be extracted from `bettrList`. This is the recommended way of calling the plotting functions, as it ensures compatibility of all components.

- **plotdata**: A data.frame with columns representing methods, metrics, scores, and weights. Typically obtained as `prepData$plotdata`, where `prepData` is the output from `bettrGetReady`.

- **scoredata**: A data.frame with columns representing methods, aggregated scores, and any other method annotations. Typically obtained as `prepData$scoredata`, where `prepData` is the output from `bettrGetReady`.

- **idCol**: Character scalar indicating which column of `plotdata` and `scoredata` contains the method IDs.
makeBarPolarPlot

**Value**

A `ggplot` object.

**Author(s)**

Charlotte Soneson

**Examples**

```r
## Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),
                 metric1 = c(1, 2, 3),
                 metric2 = c(3, 1, 2))
metricInfo <- data.frame(Metric = c("metric1", "metric2", "metric3"),
                         Group = c("G1", "G2", "G2"))
idInfo <- data.frame(Method = c("M1", "M2", "M3"),
                      Group = c("G1", "G2", "G2"))
```
makeHeatmap

Create a summary heatmap

Description

Create a summary heatmap. The input arguments for this functions are typically generated using `bettrGetReady`, which ensures that all required columns are available.

Usage

```r
makeHeatmap(
    bettrList = NULL,
    plotdata,
    scoredata,
    idCol,
    metricCol = "Metric",
    valueCol = "ScaledValue",
    weightCol = "Weight",
    scoreCol = "Score",
    metricGroupCol = "metricGroup",
    metricInfo,
    metricColors,
    idInfo,
    idColors,
    metricCollapseGroup = FALSE,
    metricGrouping = "---",
    labelSize = 10,
    showRowNames = TRUE,
    plotType = "Heatmap",
    rownamewidth_cm = 6,
    colnameheight_cm = 6
)
```

Arguments

- `bettrList`: A list, the output object from `prepData`. If `bettrList` is provided, arguments `plotdata, scoredata, idCol, metricCol, valueCol, weightCol, scoreCol, metricGroupCol, metricInfo, metricColors, idInfo, idColors, metricCollapseGroup, metricGrouping` and methods will be ignored and the corresponding values will be extracted from `bettrList`. This is the recommended way of calling the plotting functions, as it ensures compatibility of all components.
**makeHeatmap**

**plotdata**  A data.frame with columns representing methods, metrics, scores, and weights. Typically obtained as prepData$plotdata, where prepData is the output from bettrGetReady.

**scoredata**  A data.frame with columns representing methods, aggregated scores, and any other method annotations. Typically obtained as prepData$scoredata, where prepData is the output from bettrGetReady.

**idCol**  Character scalar indicating which column of plotdata and scoredata contains the method IDs.

**metricCol**  Character scalar indicating which column of plotdata contains the metric IDs. Typically, "Metric".

**valueCol**  Character scalar indicating which column of plotdata contains the metric values. Typically, "ScaledValue".

**weightCol**  Character scalar indicating which column of plotdata contains the weight values. Typically, "Weight".

**scoreCol**  Character scalar indicating which column of scoredata contains the aggregated score values. Typically, "Score".

**metricGroupCol**  Character scalar indicating which column of plotdata contains the information about the metric group. Typically, "metricGroup".

**metricInfo**  data.frame with annotations for metrics. Typically obtained as prepData$metricInfo, where prepData is the output from bettrGetReady.

**metricColors**  Named list with colors used for the metrics and any other metric annotations. Typically obtained as prepData$metricColors, where prepData is the output from bettrGetReady.

**idInfo**  data.frame with annotations for entities. Typically obtained as prepData$idInfo, where prepData is the output from bettrGetReady.

**idColors**  Named list with colors used for methods and any other method annotations. Typically obtained as prepData$idColors, where prepData is the output from bettrGetReady.

**metricCollapseGroup**  Logical scalar indicating whether metrics should be collapsed by the group variable provided by metricGrouping. Typically obtained as prepData$metricCollapseGroup, where prepData is the output from bettrGetReady.

**metricGrouping**  Character scalar indicating the column of metricInfo that was used to group metrics. Typically obtained as prepData$metricGrouping, where prepData is the output from bettrGetReady.

**labelSize**  Numeric scalar providing the size of the labels in the plot.

**showRowNames**  Logical scalar indicating whether to show row (method) names in the heatmap.

**plotType**  Either "Heatmap" or "Dot plot" indicating the type of plot to construct.

**rownamewidth_cm, colnameheight_cm**  Numeric scalars defining the width of row names and height of column names, in cm.

**Value**

A HeatmapList object.
makeParCoordPlot

Create a parallel coordinates plot

Description

Create a parallel coordinates plot. The input arguments for this function are typically generated using `bettrGetReady`, which ensures that all required columns are available.

Usage

```r
makeParCoordPlot(
  bettrList = NULL,
  plotdata,
  idCol,
  metricCol = "Metric",
  valueCol = "ScaledValue",
  metricGroupCol = "metricGroup",
  metricColors,
  idColors,
  methods = NULL,
  metricGrouping = "---",
  highlightMethod = NULL,
  labelSize = 10
)
```
Arguments

bettrList A list, the output object from prepData. If bettrList is provided, arguments plotdata, scoredata, idCol, metricCol, valueCol, weightCol, scoreCol, metricGroupCol, metricInfo, metricColors, idInfo, idColors, metricCollapseGroup, metricGrouping and methods will be ignored and the corresponding values will be extracted from bettrList. This is the recommended way of calling the plotting functions, as it ensures compatibility of all components.

plotdata A data.frame with columns representing methods, metrics, scores, and weights. Typically obtained as prepData$plotdata, where prepData is the output from bettrGetReady.

idCol Character scalar indicating which column of plotdata and scoredata contains the method IDs.

metricCol Character scalar indicating which column of plotdata contains the metric IDs. Typically, "Metric".

valueCol Character scalar indicating which column of plotdata contains the metric values. Typically, "ScaledValue".

metricGroupCol Character scalar indicating which column of plotdata contains the information about the metric group. Typically, "metricGroup".

metricColors Named list with colors used for the metrics and any other metric annotations. Typically obtained as prepData$metricColors, where prepData is the output from bettrGetReady.

idColors Named list with colors used for methods and any other method annotations. Typically obtained as prepData$idColors, where prepData is the output from bettrGetReady.

methods Character vector containing the methods to include. If NULL (default), all methods will be used.

metricGrouping Character scalar indicating the column of metricInfo that was used to group metrics. Typically obtained as prepData$metricGrouping, where prepData is the output from bettrGetReady.

highlightMethod Character scalar indicating a method that should be highlighted in the plot.

labelSize Numeric scalar providing the size of the labels in the plot.

Value

A ggplot object.

Author(s)

Charlotte Soneson

Examples

## Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

### Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),

```r
metric1 = c(1, 2, 3),
metric2 = c(3, 1, 2))
metricInfo <- data.frame(Metric = c("metric1", "metric2", "metric3"),
                         Group = c("G1", "G2", "G2"))
idInfo <- data.frame(Method = c("M1", "M2", "M3"),
                     Type = c("T1", "T1", "T2"))
prepData <- bettrGetReady(df = df, idCol = "Method",
                          metricInfo = metricInfo, idInfo = idInfo)
makeParCoordPlot(bettrList = prepData, highlightMethod = "M2")
```

---

### makePolarPlot

Create a polar plot. The input arguments for this function are typically generated using `bettrGetReady`, which ensures that all required columns are available.

#### Usage

```r
makePolarPlot(
  bettrList = NULL,
  plotdata,
  idCol,
  metricCol = "Metric",
  valueCol = "ScaledValue",
  metricGroupCol = "metricGroup",
  metricColors,
  metricCollapseGroup = FALSE,
  metricGrouping = "---",
  labelSize = 10
)
```

#### Arguments

- **bettrList**: A list, the output object from `prepData`. If `bettrList` is provided, arguments `plotdata`, `scoredata`, `idCol`, `metricCol`, `valueCol`, `weightCol`, `scoreCol`, `metricGroupCol`, `metricInfo`, `metricColors`, `idInfo`, `idColors`, `metricCollapseGroup`, `metricGrouping` and methods will be ignored and the corresponding values will be extracted from `bettrList`. This is the recommended way of calling the plotting functions, as it ensures compatibility of all components.

- **plotdata**: A `data.frame` with columns representing methods, metrics, scores, and weights. Typically obtained as `prepData$plotdata`, where `prepData` is the output from `bettrGetReady`.

- **idCol**: Character scalar indicating which column of `plotdata` and `scoredata` contains the method IDs.
**makePolarPlot**

**metricCol** Character scalar indicating which column of plotdata contains the metric IDs. Typically, "Metric".

**valueCol** Character scalar indicating which column of plotdata contains the metric values. Typically, "ScaledValue".

**metricGroupCol** Character scalar indicating which column of plotdata contains the information about the metric group. Typically, "metricGroup".

**metricColors** Named list with colors used for the metrics and any other metric annotations. Typically obtained as prepData$metricColors, where prepData is the output from bettrGetReady.

**metricCollapseGroup** Logical scalar indicating whether metrics should be collapsed by the group variable provided by metricGrouping. Typically obtained as prepData$metricCollapseGroup, where prepData is the output from bettrGetReady.

**metricGrouping** Character scalar indicating the column of metricInfo that was used to group metrics. Typically obtained as prepData$metricGrouping, where prepData is the output from bettrGetReady.

**labelSize** Numeric scalar providing the size of the labels in the plot.

**Value**

A ggplot object.

**Author(s)**

Charlotte Soneson

**Examples**

```r
## Generate example data
df <- data.frame(Method = c("M1", "M2", "M3"),
                 metric1 = c(1, 2, 3),
                 metric2 = c(3, 1, 2))
metricInfo <- data.frame(Metric = c("metric1", "metric2", "metric3"),
                         Group = c("G1", "G2", "G2"))
idInfo <- data.frame(Method = c("M1", "M2", "M3"),
                     Type = c("T1", "T1", "T2"))
prepData <- bettrGetReady(df = df, idCol = "Method",
                           metricInfo = metricInfo, idInfo = idInfo)
makePolarPlot(bettrList = prepData)
```
Index

* internal
  bettr-package, 2

assembleSE, 3

bettr, 5
bettr-package, 2
bettrGetReady, 7, 10, 12, 14, 16

HeatmapList, 13

makeBarPolarPlot, 10
makeHeatmap, 12
makeParCoordPlot, 14
makePolarPlot, 16