Package: CytoProfile (via r-universe)

October 9, 2026

Title Cytokine Profiling Analysis Tool

Version 0.2.4

Description Provides comprehensive cytokine profiling analysis through quality control using biologically meaningful cutoffs on raw cytokine measurements and by testing for distributional symmetry to recommend appropriate transformations. Offers exploratory data analysis with summary statistics, enhanced boxplots, and barplots, along with univariate and multivariate analytical capabilities for in-depth cytokine profiling such as Principal Component Analysis based on Andrzej Maćkiewicz and Waldemar Ratajczak (1993) <doi:10.1016/0098-3004(93)90090-R>, Sparse Partial Least Squares Discriminant Analysis based on Lê Cao K-A, Boitard S, and Besse P (2011) <doi:10.1186/1471-2105-12-253>, Random Forest based on Breiman, L. (2001) <doi:10.1023/A:1010933404324>, and Extreme Gradient Boosting based on Tianqi Chen and Carlos Guestrin (2016) <doi:10.1145/2939672.2939785>.

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL https://github.com/saraswatsh/CytoProfile, https://cytoprofile.cytokineprofile.org/

Depends R (>= 4.3)

Imports mixOmics, dplyr, tidyr, pROC, plot3D, caret, xgboost, randomForest, pheatmap, e1071, ggplot2, ggrepel, gridExtra, reshape2, lifecycle, data.table

Suggests spelling, BiocManager, Ckmeans.1d.dp, prodlim, testthat, knitr, rmarkdown, devtools

NeedsCompilation no

License GPL (>= 2)

LazyData true

VignetteBuilder knitr

BugReports https://github.com/saraswatsh/CytoProfile/issues

Language en-US

Config/pak/sysreqs cmake libfreetype6-dev libglpk-dev libglu1-mesa-dev make texlive libicu-dev libpng-dev libuv1-dev libxml2-dev libgl1-mesa-dev zlib1g-dev

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

Date/Publication 2026-02-27 17:54:17 UTC

RemoteUrl https://github.com/saraswatsh/cytoprofile

RemoteRef HEAD

RemoteSha ba676b9ef0fa79ce42917523ec23dd518e7436e6

Contents

adjust_p . . . 3
apply_scale . . . 3
cyt_anova . . . 4
cyt_bp . . . 5
cyt_bp2 . . . 6
cyt_dualflashplot . . . 7
cyt_errbp . . . 8
cyt_export . . . 10
cyt_heatmap . . . 11
cyt_mint_splsda . . . 13
cyt_pca . . . 15
cyt_rf . . . 17
cyt_skku . . . 19
cyt_splsda . . . 21
cyt_ttest . . . 23
cyt_univariate . . . 25
cyt_univariate_multi . . . 26
cyt_violin . . . 27
cyt_volc . . . 28
cyt_xgb . . . 30
ExampleData1 . . . 32
ExampleData2 . . . 34
ExampleData3 . . . 35
ExampleData4 . . . 36
ExampleData5 . . . 37

Index **39**

adjust_p *Adjust p-values using a specified method*

Description

A thin wrapper around `stats::p.adjust` that defaults to the Benjamini–Hochberg procedure. Useful for unifying multiple testing adjustments across the package.

Usage

```
adjust_p(p_values, method = "BH")
```

Arguments

`p_values`	A numeric vector of raw p-values.
`method`	A character string specifying the p-value adjustment method. Passed directly to `p.adjust`. Defaults to "BH" (Benjamini–Hochberg). See `p.adjust.methods` for other options.

Value

A numeric vector of adjusted p-values of the same length as `p_values`.

apply_scale *Apply a scale transformation to numeric columns*

Description

This helper function applies a chosen scaling or transformation to specified numeric columns in a data frame. Supported built-in transformations include no transformation ("none"), log2, log10, and z-score scaling. A custom function can also be supplied to perform arbitrary transformations.

Usage

```
apply_scale(
  data,
  columns = NULL,
  scale = c("none", "log2", "log10", "zscore", "custom"),
  custom_fn = NULL
)
```

Arguments

data	A data.frame or matrix containing the data to be transformed.
columns	A character vector of column names to transform. If NULL (default) all numeric columns will be transformed.
scale	A character string specifying the transformation to apply. Possible values are "none", "log2", "log10", "zscore", or "custom". When set to "custom" the function specified in custom_fn will be applied to the columns.
custom_fn	A function that takes a numeric vector and returns a transformed numeric vector. Only used when scale = "custom".

Value

A data.frame with the same dimensions as data with transformed numeric columns.

---

cyt_anova *ANOVA Analysis on Continuous Variables.* **[Deprecated]**

Description

This function performs an analysis of variance (ANOVA) for each continuous variable against every categorical predictor in the input data. Character columns are automatically converted to factors; all factor columns are used as predictors while numeric columns are used as continuous outcomes. For each valid predictor (i.e., with more than one level and no more than 10 levels), Tukey's Honest Significant Difference (HSD) test is conducted and the adjusted p-values for pairwise comparisons are extracted.

Usage

```
cyt_anova(data, format_output = FALSE)
```

Arguments

data	A data frame or matrix containing both categorical and continuous variables. Character columns will be converted to factors and used as predictors, while numeric columns will be used as continuous outcomes.
format_output	Logical. If TRUE, returns the results as a tidy data frame instead of a list. Default is FALSE.

Value

If format_output is FALSE (default), a list of adjusted p-values from Tukey's HSD tests for each combination of continuous outcome and categorical predictor. List elements are named in the format "Outcome_Categorical". If format_output is TRUE, a data frame in a tidy format.

Author(s)

Shubh Saraswat

Examples

```
data("ExampleData1")
cyt_anova(ExampleData1[, c(1:2, 5:6)], format_output = TRUE)
```

---

cyt_bp *Boxplots for Continuous Variables with Optional Grouping*

Description

This function generates boxplots for numeric variables in a data frame or matrix. It supports optional grouping by one or more categorical variables. Numeric variables can be scaled using various transformations before plotting. When grouping is not used, boxplots are arranged in pages with a specified maximum number of plots per page. Plots can be saved to a PDF file or displayed on the current graphics device.

Usage

```
cyt_bp(
  data,
  output_file = NULL,
  group_by = NULL,
  bin_size = 25,
  y_lim = NULL,
  scale = c("none", "log2", "log10", "zscore", "custom"),
  custom_fn = NULL
)
```

Arguments

data	A matrix or data frame containing numeric and categorical variables.
output_file	Optional string specifying the name of the file to be created. When NULL (default), plots are drawn on the current graphics device. Ensure that the file extension matches the desired format (e.g., ".pdf" for PDF output or ".png" for PNG output or .tiff for TIFF output).
group_by	Optional character vector specifying one or more columns to use for grouping. If NULL (default) no grouping is applied.
bin_size	Integer. Maximum number of boxplots per page when grouping is not used. Default is 25, as in the original cyt_bp.
y_lim	Optional numeric vector giving y-axis limits for the plots. Applies to all plots.
scale	Character specifying a transformation for numeric variables. Accepts "none", "log2", "log10", "zscore", or "custom". When "custom", supply a function via custom_fn.
custom_fn	A user supplied function to transform numeric columns when scale = "custom".

Value

Invisibly returns a list of ggplot objects. When output_file is provided, plots are written to the PDF file.

Examples

```
data("ExampleData1")
# Boxplots without grouping
cyt_bp(ExampleData1[, -c(1:3)], output_file = NULL, scale = "log2")
# Boxplots grouped by Group
cyt_bp(ExampleData1[, -c(3,5:28)], group_by = "Group", scale = "zscore")
```

---

cyt_bp2 *Boxplot Function Enhanced for Specific Group Comparisons.* **[Deprecated]**

Description

This function generates a PDF file containing boxplots for each combination of numeric and factor variables in the provided data. It first converts any character columns to factors and checks that the data contains at least one numeric and one factor column. If the scale argument is set to "log2", all numeric columns are log2-transformed. The function then creates boxplots using ggplot2 for each numeric variable grouped by each factor variable.

Usage

```
cyt_bp2(data, pdf_title, scale = NULL, y_lim = NULL)
```

Arguments

data	A matrix or data frame of raw data.
pdf_title	A string representing the title (and filename) of the PDF file. If NULL, the boxplots are displayed on the current graphics device. Defaults to NULL.
scale	Transformation option for continuous variables. Options are NULL (default) and "log2". When set to "log2", numeric columns are transformed using the log2 function.
y_lim	An optional numeric vector defining the y-axis limits for the plots.

Value

A PDF file containing the boxplots.

Author(s)

Shubh Saraswat

Examples

```
# Loading data
data_df <- ExampleData1[, -c(3, 5:28)]
data_df <- dplyr::filter(data_df, Group == "T2D", Treatment == "Unstimulated")
cyt_bp2(data_df, pdf_title = NULL, scale = "log2")
```

`cyt_dualflashplot` *Dual-flashlight Plot*

Description

This function reshapes the input data and computes summary statistics (mean and variance) for each variable grouped by a specified factor column. It then calculates the SSMD (Strictly Standardized Mean Difference) and log2 fold change between two groups (group1 and group2) and categorizes the effect strength as "Strong Effect", "Moderate Effect", or "Weak Effect". A dual flash plot is generated using ggplot2 where the x-axis represents the average log2 fold change and the y-axis represents the SSMD. Additionally, the function prints the computed statistics to the console. A ggplot object is returned for further modification.

Usage

```
cyt_dualflashplot(
  data,
  group_var,
  group1,
  group2,
  ssmd_thresh = 1,
  log2fc_thresh = 1,
  top_labels = 15,
 category_labels = c(Strong = "Strong Effect", Moderate = "Moderate Effect", Weak =
    "Weak Effect"),
 colors = c(`Strong Effect` = "red", `Moderate Effect` = "orange", `Weak Effect` =
    "blue"),
  shapes = c(`FALSE` = 16, `TRUE` = 17),
  verbose = FALSE
)
```

Arguments

`data`	A data frame containing at least one numeric column and a grouping column.
`group_var`	Character. Name of the grouping column.
`group1`	Character strings identifying the two levels of `group_var` to compare.
`group2`	Character strings identifying the two levels of `group_var` to compare.
`ssmd_thresh`	Numeric. Absolute SSMD threshold for highlighting observations as significant. Default is 1.

log2fc_thresh	Numeric. Absolute log2 fold change threshold for significance. Default is 1.
top_labels	Integer. Number of variables with the largest absolute SSMD to label on the plot. Default is 15.
category_labels	Optional named character vector of length three providing alternative labels for the SSMD effect categories. Names must include "Strong", "Moderate" and "Weak". Defaults are `c(Strong = "Strong Effect", Moderate = "Moderate Effect", Weak = "Weak Effect")`.
colors	Optional named character vector mapping the effect categories to colors. Names must match those in `category_labels`. Defaults are red for strong, orange for moderate and blue for weak.
shapes	Optional named numeric vector of length two giving the plotting characters for non-significant and significant points. Defaults are `c(FALSE= 16, TRUE = 17)` (solid and triangular symbols).
verbose	Logical. If `TRUE`, prints the computed summary statistics. Default is `FALSE`.

Value

A ggplot object representing the dual-flashlight plot. When `verbose` is `TRUE`, the summary statistics data frame is printed to the console.

Author(s)

Xiaohua Douglas Zhang and Shubh Saraswat

Examples

```
# Loading data
data_df <- ExampleData1[, -c(2:3)]
cyt_dualflashplot(data_df, group_var = "Group", group1 = "T2D", group2 = "ND",
                        ssmd_thresh = 0.2, log2fc_thresh = 1,
                        top_labels = 10)
```

---

cyt_errbp *Error-bar Plot*

Description

This function generates an error-bar plot to visually compare different groups against a designated baseline group. It displays the central tendency (mean or median) as a bar and overlays error bars to represent the data's spread (e.g., standard deviation, MAD, or standard error). The plot can also include p-value and effect size labels (based on SSMD), presented either as symbols or numeric values, to highlight significant differences and the magnitude of effects. When an output filename is provided the plot is saved to disk; otherwise the ggplot object is returned and drawn on the current graphics device.

Usage

```
cyt_errbp(
  data,
  group_col = NULL,
  p_lab = TRUE,
  es_lab = TRUE,
  class_symbol = FALSE,
  x_lab = "",
  y_lab = "",
  title = "",
  stat = c("mean", "median"),
  error = c("se", "sd", "mad", "ci"),
  scale = c("none", "log2", "log10", "zscore", "custom"),
  custom_fn = NULL,
  method = c("auto", "ttest", "wilcox"),
  p_adjust_method = NULL,
  output_file = NULL,
  label_size = 4
)
```

Arguments

data	A data frame containing at least one numeric column and a grouping column.
group_col	Character string naming the column that defines groups. This column will be coerced to a factor.
p_lab	Logical. If TRUE (default) p-value labels are displayed for group comparisons.
es_lab	Logical. If TRUE (default) effect-size labels are displayed.
class_symbol	Logical. If TRUE, p-value and effect-size labels are encoded using symbols (e.g., *, >>>). If FALSE(default), numeric values are shown instead.
x_lab	Character string for the x-axis label. If empty a default label is generated.
y_lab	Character string for the y-axis label. If empty a default label is generated.
title	Character string for the plot title. If empty a default title is generated.
stat	Character. Central tendency statistic to use. Choices are "mean" or "median"; default is "mean". Added to support non-mean summaries.
error	Character. Error measure visualized around the statistic. Options are "se" (standard error; default), "sd" (standard deviation), "mad" (median absolute deviation) or "ci" (approximate 95 % confidence interval).
scale	Character controlling data transformation before analysis. Accepts "none" (default), "log2", "log10", "zscore" or "custom".
custom_fn	A user-supplied function applied to numeric columns when scale = "custom".
method	Character controlling the statistical test used for pairwise comparisons. Options are "auto" (default; choose between t-test and Wilcoxon based on a normality test), "ttest" or "wilcox".

p_adjust_method
: Character. If non-NULL, specifies the method used by `p.adjust()` to correct p-values across all comparisons (e.g., "BH" for Benjamini–Hochberg). If `NULL` (default) no adjustment is performed.

output_file
: Optional file path. If provided, the plot is saved using `ggsave()`; otherwise the plot is returned and automatically printed.

label_size
: Numeric. Font size for p-value and effect-size labels. Default is 4.

Value

An error-bar plot (a `ggplot` object) is produced and optionally saved as a PDF. If `output_file` is specified, the function returns returns the `ggplot` object.

Author(s)

Xiaohua Douglas Zhang and Shubh Saraswat

Examples

```
# Basic usage with default settings
df <- ExampleData1[, c("Group", "CCL-20/MIP-3A", "IL-10")]
cyt_errbp(df, group_col = "Group")
# Use mean and SD, log2 transform and show significance
cyt_errbp(df, group_col = "Group", stat = "mean", error = "sd",
          scale = "log2", class_symbol = TRUE, method = "ttest")
```

---

cyt_export *Generic export function for Cytokine plots*

Description

The purpose of this helper is to save a collection of plots generated throughout the CytoProfile package to a user specified format. It is designed to handle objects created with `ggplot2` (or other grid based plots), recorded base plots (see `grDevices::recordPlot()`), or custom drawing functions stored as closures. If a multi-page PDF is requested, all plots are written into a single file; for raster formats (PNG, JPEG and TIFF), each element of the plot list is saved to its own file with a numbered suffix. Nothing is returned except invisibly.

Usage

```
cyt_export(
  plots,
  filename = "cyto_output",
  format = c("pdf", "png", "jpeg", "tiff", "svg"),
  width = 7,
  height = 5,
  dpi = 300,
  which = NULL
)
```

Arguments

plots	A single plot object or a list of plots. Each element may be a ggplot/grid object, a recordedplot, or a function with no arguments that when invoked produces the plot.
filename	Base file name (without extension). The appropriate extension is added automatically depending on format. A full path can be supplied to save into a particular directory. When saving raster formats the file names will include an index (e.g. "myplot_001.png").
format	Character string giving the desired output format. One of "pdf", "png", "jpeg" or "tiff". Partial matching is performed. The default is "pdf".
width, height	Width and height of the device in inches. These arguments are passed directly to the graphics device.
dpi	Resolution in dots per inch used for raster formats (ignored for PDF).
which	Optional character vector naming the subset of plots to save. If provided, only the named plots are exported; otherwise all elements are saved in the order they appear.

Examples

```
# Example usage within CytoProfile:
# res <- cyt_splsda(...)
# cyt_export(res$plots, filename = "splsda", format = "png")
```

---

cyt_heatmap *Heat Map*

Description

This function creates a heatmap using the numeric columns from the provided data frame. It provides the ability to hide row and column names, adjust font sizes and clustering, and apply additional transformations such as log10 or combined z-scoring. A file name with extension may be provided via title to save the heat map to disk; otherwise the plot is drawn on the active graphics device.

Usage

```
cyt_heatmap(
  data,
  scale = c(NULL, "log2", "log10", "row_zscore", "col_zscore", "zscore"),
  annotation_col = NULL,
  annotation_side = c("auto", "row", "col"),
  show_row_names = FALSE,
  show_col_names = TRUE,
  fontsize_row = 10,
  fontsize_col = 10,
```

```
  cluster_rows = TRUE,
  cluster_cols = TRUE,
  title = NULL
)
```

Arguments

data	A data frame. Only numeric columns are used to construct the heat map.
scale	Character specifying an optional scaling. Accepts NULL (no scaling), "log2", "log10", "row_zscore", "col_zscore" or "zscore" (apply both row and column z-scoring). Default is NULL.
annotation_col	Optional. Either the name of a column in data or a vector of length equal to the number of rows or columns of the numeric matrix. If a column name is supplied the function determines whether it annotates rows or columns based on its length or the value of annotation_side.
annotation_side	Character. One of "auto", "row" or "col". When "auto" (default) the side is determined by matching the length of annotation_col to rows or columns.
show_row_names	Logical. If TRUE row names are shown Default is FALSE.
show_col_names	Logical. If FALSE column names are hidden. Default is TRUE.
fontsize_row	Numeric. Font size for row names. Default is 10.
fontsize_col	Numeric. Font size for column names. Default is 10.
cluster_rows	Logical. If TRUE (default), rows are clustered.
cluster_cols	Logical. If TRUE (default), columns are clustered.
title	Character. The heat map title or file name. If title ends with ".pdf" or ".png" (case insensitive), the heat map is saved to that file and no title is printed on screen. If NULL (default), the heat map is drawn on the active device without saving and without a main title.

Value

Invisibly returns the pheatmap object created by pheatmap::pheatmap().

Author(s)

Shubh Saraswat

Examples

```
# Load sample data
data("ExampleData1")
data_df <- ExampleData1
# Generate a heatmap with log2 scaling and annotation based on
# the "Group" column
cyt_heatmap(
  data = data_df[, -c(2:3)],
  scale = "log2",  # Optional scaling
```

```
  annotation_col = "Group",
  annotation_side = "auto",
  title = NULL
)
```

`cyt_mint_splsda` *Analyze data with Multivariate INTegration (MINT) Sparse Partial Least Squares Discriminant Analysis (sPLS-DA).*

Description

This function performs a MINT (Multivariate INTegrative) sPLS-DA to handle batch effects by modeling a global biological signal across different studies or batches. If a second grouping column (`group_col2`) is provided, the analysis is stratified and performed for each level of that column.

Usage

```
cyt_mint_splsda(
  data,
  group_col,
  batch_col,
  group_col2 = NULL,
  colors = NULL,
  output_file = NULL,
  ellipse = TRUE,
  bg = FALSE,
  var_num = 20,
  comp_num = 2,
  scale = c("none", "log2", "log10", "zscore", "custom"),
  custom_fn = NULL,
  tune = FALSE,
  cim = FALSE,
  roc = FALSE,
  verbose = FALSE
)
```

Arguments

`data`	A matrix or data frame containing the variables. Columns not specified by `group_col`, `group_col2`, or `multilevel_col` are assumed to be continuous variables for analysis.
`group_col`	A string specifying the first grouping column name that contains grouping information. If `group_col2` is not provided, it will be used for both grouping and treatment.
`batch_col`	A string specifying the column that identifies the batch or study for each sample.
`group_col2`	A string specifying the second grouping column name. Default is `NULL`.

colors	A vector of splsda_colors for the groups or treatments. If NULL, a random palette (using rainbow) is generated based on the number of groups.
output_file	Optional string specifying the name of the file to be created. When NULL (default), plots are drawn on the current graphics device. Ensure that the file extension matches the desired format (e.g., ".pdf" for PDF output or ".png" for PNG output or .tiff for TIFF output).
ellipse	Logical. Whether to draw a 95\ Default is FALSE.
bg	Logical. Whether to draw the prediction background in the figures. Default is FALSE.
var_num	Numeric. The number of variables to be used in the PLS-DA model.
comp_num	Numeric. The number of components to calculate in the sPLS-DA model. Default is 2.
scale	Character string specifying a transformation to apply to the numeric predictor columns prior to model fitting. Options are "none", "log2", "log10", "zscore", or "custom". When "custom" is selected a user defined function must be supplied via custom_fn. Defaults to "none".
custom_fn	A custom transformation function used when scale = "custom". Ignored otherwise. It should take a numeric vector and return a numeric vector of the same length.
tune	Logical. If TRUE, performs tuning of ncomp and keepX via cross-validation. Default is FALSE.
cim	Logical. Whether to compute and plot the Clustered Image Map (CIM) heatmap. Default is FALSE.
roc	Logical. Whether to compute and plot the ROC curve for the model. Default is FALSE.
verbose	A logical value indicating whether to print additional informational output to the console. When TRUE, the function will display progress messages, and intermediate results when FALSE (the default), it runs quietly.

Details

When verbose is set to TRUE, additional information about the analysis and confusion matrices are printed to the console. These can be suppressed by keeping verbose = FALSE.

Value

Plots consisting of the classification figures, ROC curves, correlation circle plots, and heatmaps.

Author(s)

Shubh Saraswat

References

Rohart F, Eslami A, Matigian, N, Bougeard S, Lê Cao K-A (2017). MINT: A multivariate integrative approach to identify a reproducible biomarker signature across multiple experiments and platforms. BMC Bioinformatics 18:128.

Examples

```
# Loading ExampleData5 dataset with batch column
data_df <- ExampleData5[,-c(2,4)]
data_df <- dplyr::filter(data_df, Group != "ND")

cyt_mint_splsda(data_df, group_col = "Group",
 batch_col = "Batch", colors = c("black", "purple"),
 ellipse = TRUE, var_num = 25, comp_num = 2,
 scale = "log2", verbose = FALSE)
```

`cyt_pca` *Analyze Data with Principal Component Analysis (PCA) for Cytokines.*

Description

This function performs Principal Component Analysis (PCA) on cytokine data and generates several types of plots, including:

- 2D PCA plots using mixOmics' `plotIndiv` function,
- 3D scatter plots (if `style` is "3d" or "3D" and `comp_num` is 3) via the plot3D package,
- Scree plots showing both individual and cumulative explained variance,
- Loadings plots, and
- Biplots and correlation circle plots.

Usage

```
cyt_pca(
  data,
  group_col = NULL,
  group_col2 = NULL,
  colors = NULL,
  output_file,
  ellipse = FALSE,
  comp_num = 2,
  scale = c("none", "log2", "log10", "zscore", "custom"),
  custom_fn = NULL,
  pch_values = NULL,
  style = NULL
)
```

Arguments

data	A data frame containing cytokine data. It should include at least one column representing grouping information and optionally a second column representing treatment or stimulation.
group_col	A string specifying the column name that contains the first group information. If group_col2 is not provided, an overall analysis will be performed.
group_col2	A string specifying the second grouping column. Default is NULL.
colors	A vector of colors corresponding to the groups. If set to NULL, a palette is generated using rainbow() based on the number of unique groups.
output_file	Optional string specifying the name of the file to be created. When NULL (default), plots are drawn on the current graphics device. Ensure that the file extension matches the desired format (e.g., ".pdf" for PDF output or ".png" for PNG output or .tiff for TIFF output).
ellipse	Logical. If TRUE, a 95% confidence ellipse is drawn on the PCA individuals plot. Default is FALSE.
comp_num	Numeric. The number of principal components to compute and display. Default is 2.
scale	Character string specifying a transformation to apply to numeric variables before PCA. Options are "none" (no transformation), "log2", "log10", "zscore", or "custom". When "custom" is selected, a user supplied function must be given via custom_fn. Defaults to "none".
custom_fn	A custom function used when scale = "custom". Should take a numeric vector and return a numeric vector. Ignored otherwise.
pch_values	A vector of plotting symbols (pch values) to be used in the PCA plots. Default is NULL.
style	Character. If set to "3d" or "3D" and comp_num equals 3, a 3D scatter plot is generated using the plot3D package. Default is NULL.

Value

A PDF file containing the PCA plots is generated and saved when output_file is provided. Otherwise, plots are displayed on the current graphics device.

Author(s)

Shubh Saraswat

Examples

```
# Load sample data
data <- ExampleData1[, -c(3,23)]
data_df <- dplyr::filter(data, Group != "ND" & Treatment != "Unstimulated")
# Run PCA analysis and save plots to a PDF file
cyt_pca(
  data = data_df,
  output_file = NULL,
```

```
  colors = c("black", "red2"),
  scale = "log2",
  comp_num = 3,
  pch_values = c(16, 4),
  style = "3D",
  group_col = "Group",
  group_col2 = "Treatment",
  ellipse = FALSE
)
```

---

cyt_rf *Run Random Forest Classification on Cytokine Data,*

Description

This function trains and evaluates a Random Forest classification model on cytokine data. It includes feature importance visualization, cross- validation for feature selection, and performance metrics such as accuracy, sensitivity, and specificity. Optionally, for binary classification, the function also plots the ROC curve and computes the AUC.

Usage

```
cyt_rf(
  data,
  group_col,
  ntree = 500,
  mtry = 5,
  train_fraction = 0.7,
  plot_roc = FALSE,
  k_folds = 5,
  step = 0.5,
  run_rfcv = TRUE,
  verbose = FALSE,
  seed = 123,
  cv = FALSE,
  cv_folds = 5,
  scale = c("none", "log2", "log10", "zscore", "custom"),
  custom_fn = NULL
)
```

Arguments

data	A data frame containing the cytokine measurements. One column should correspond to the grouping variable (the outcome) and the remaining columns should be numeric predictors.
group_col	A string naming the column in data that contains the grouping variable.

ntree	Integer specifying the number of trees to grow. Default is 500.
mtry	Integer specifying the number of variables randomly sampled at each split. Default is 5.
train_fraction	Numeric between 0 and 1 giving the proportion of data used for training. The remainder is used for testing. Default is 0.7.
plot_roc	Logical. If TRUE and the problem is binary, an ROC curve and AUC will be computed and plotted for the test set. Default is FALSE.
k_folds	Integer specifying the number of folds for rfcv when run_rfcv = TRUE. Default is 5.
step	Numeric specifying the fraction of variables removed at each step during rfcv. Default is 0.5.
run_rfcv	Logical indicating whether to run Random Forest cross-validation for feature selection. Default is TRUE.
verbose	Logical indicating whether to print intermediate results. When TRUE, training and test performance metrics, confusion matrices and cross-validation details are printed. Default is FALSE.
seed	Optional integer seed for reproducibility. Default is 123.
cv	Logical indicating whether to perform a separate k-fold classification cross-validation using caret. Default is FALSE.
cv_folds	Integer specifying the number of folds for classification cross-validation when cv = TRUE. Default is 5.
scale	Character string specifying a transformation to apply to the numeric predictor columns prior to model fitting. Options are "none" (no transformation), "log2", "log10", "zscore", or "custom". When "custom" is selected a user defined function must be supplied via custom_fn. Defaults to "none".
custom_fn	A custom transformation function used when scale = "custom". The function should take a numeric vector and return a numeric vector of the same length. Ignored for other values of scale.

Details

The function first coerces the grouping variable to a factor and splits the dataset into training and test subsets according to train_fraction. A Random Forest classifier is fit to the training data using the specified ntree and mtry parameters. The model performance is assessed on both the training and test sets, and results are printed when verbose = TRUE. If plot_roc = TRUE and the grouping variable has exactly two levels, an ROC curve is computed on the test set and a plot is returned. Variable importance is extracted and visualized with a bar plot. Optionally, cross- validation for feature selection (rfcv) is performed and the error curve is plotted. A separate k-fold classification cross- validation using caret::train can be requested via cv = TRUE.

Value

An invisible list with components:

model	The fitted randomForest model.

confusion_matrix
: Confusion matrix on the test set.

importance_plot
: A ggplot2 object of the variable importance (mean decrease in Gini).

importance_data
: A data frame of variable importance values.

rfcv_result
: The rfcv object returned when run_rfcv = TRUE.

rfcv_plot
: A ggplot2 object of cross-validation error versus number of variables, returned when run_rfcv = TRUE.

rfcv_data
: A data frame summarizing the rfcv error curve.

roc_plot
: A ggplot2 object of the ROC curve for binary classification when plot_roc = TRUE.

cv_results
: A caret train object returned when cv = TRUE or NULL otherwise.

Examples

```
data.df0 <- ExampleData1
data.df <- data.frame(data.df0[, 1:3], log2(data.df0[, -c(1:3)]))
data.df <- data.df[, -c(2:3)]
data.df <- dplyr::filter(data.df, Group != "ND")

cyt_rf(
  data = data.df, group_col = "Group", k_folds = 5, ntree = 1000,
  mtry = 4, run_rfcv = TRUE, plot_roc = TRUE, verbose = FALSE
)
```

---

cyt_skku *Distribution of the Data Set Shown by Skewness and Kurtosis.*

Description

This function computes summary statistics — including sample size, mean, standard error, skewness, and kurtosis — for each numeric measurement column in a data set. If grouping columns are provided via group_cols, the function computes the metrics separately for each group defined by the combination of these columns (using the first element as the treatment variable and the second as the grouping variable, or the same column for both if only one is given). When no grouping columns are provided, the entire data set is treated as a single group ("Overall"). A log2 transformation (using a cutoff equal to one-tenth of the smallest positive value in the data) is applied to generate alternative metrics. Histograms showing the distribution of skewness and kurtosis for both raw and log2-transformed data are then generated and saved to a PDF if a file name is provided.

Usage

```
cyt_skku(
  data,
  group_cols = NULL,
```

```
  output_file = NULL,
  print_res_raw = FALSE,
  print_res_log = FALSE
)
```

Arguments

data	A matrix or data frame containing the raw data. If group_cols is specified, the columns with names in group_cols are treated as grouping variables and all other columns are assumed to be numeric measurement variables.
group_cols	A character vector specifying the names of the grouping columns. When provided, the first element is treated as the treatment variable and the second as the group variable. If not provided, the entire data set is treated as one group.
output_file	Optional string specifying the name of the file to be created. When NULL (default), plots are drawn on the current graphics device. Ensure that the file extension matches the desired format (e.g., ".pdf" for PDF output or ".png" for PNG output or .tiff for TIFF output).
print_res_raw	Logical. If TRUE, the function returns and prints the computed summary statistics for the raw data. Default is FALSE.
print_res_log	Logical. If TRUE, the function returns and prints the computed summary statistics for the log2-transformed data. Default is FALSE.

Details

A cutoff is computed as one-tenth of the minimum positive value among all numeric measurement columns to avoid taking logarithms of zero. When grouping columns are provided, the function loops over unique grouping columns and computes the metrics for each measurement column within each subgroup. Without grouping columns, the entire data set is analyzed as one overall group.

Value

The function generates histograms of skewness and kurtosis for both raw and log2-transformed data. Additionally, if either printResRaw and/or printResLog is TRUE, the function returns the corresponding summary statistics as a data frame or a list of data frames.

Author(s)

Xiaohua Douglas Zhang and Shubh Saraswat

Examples

```
# Example with grouping columns (e.g., "Group" and "Treatment")
data(ExampleData1)
cyt_skku(ExampleData1[, -c(2:3)], output_file = NULL,
  group_cols = c("Group")
)

# Example without grouping columns (analyzes the entire data set)
cyt_skku(ExampleData1[, -c(1:3)], output_file = NULL)
```

cyt_splsda *Analyze data with Sparse Partial Least Squares Discriminant Analysis (sPLS-DA).*

Description

This function conducts Sparse Partial Least Squares Discriminant Analysis (sPLS-DA) on the provided data. It uses the specified `group_col` (and optionally `group_col2`) to define class labels while assuming the remaining columns contain continuous variables. The function supports transformations via the `scale` parameter and generates a series of plots, including classification plots, scree plots, loadings plots, and VIP score plots. Optionally, ROC curves are produced when `roc` is `TRUE`. Additionally, cross-validation is supported via LOOCV or Mfold methods. When both `group_col` and `group_col2` are provided and differ, the function analyzes each treatment level separately.

Usage

```
cyt_splsda(
  data,
  group_col = NULL,
  group_col2 = NULL,
  multilevel_col = NULL,
  batch_col = NULL,
  ind_names = FALSE,
  colors = NULL,
  output_file = NULL,
  ellipse = FALSE,
  bg = FALSE,
  conf_mat = FALSE,
  var_num,
  cv_opt = NULL,
  fold_num = 5,
  scale = c("none", "log2", "log10", "zscore", "custom"),
  custom_fn = NULL,
  tune = FALSE,
  tune_folds = 5,
  comp_num = 2,
  pch_values,
  style = NULL,
  roc = FALSE,
  verbose = FALSE,
  seed = 123
)
```

Arguments

`data`	A matrix or data frame containing the variables. Columns not specified by `group_col` or `group_col2` are assumed to be continuous variables for analysis.

group_col	A string specifying the column name that contains the first group information. If group_col2 is not provided, an overall analysis will be performed.
group_col2	A string specifying the second grouping column. Default is NULL.
multilevel_col	A string specifying the column name that identifies repeated measurements (e.g., patient or sample IDs). If provided, a multilevel analysis will be performed. Default is NULL.
batch_col	A string specifying the column that identifies the batch or study for each sample.
ind_names	If TRUE, the row names of the first (or second) data matrix is used as names. Default is FALSE. If a character vector is provided, these values will be used as names. If 'pch' is set this will overwrite the names as shapes. See ?mixOmics::plotIndiv for details.
colors	A vector of colors for the groups or treatments. If NULL, a random palette (using rainbow) is generated based on the number of groups.
output_file	Optional string specifying the name of the file to be created. When NULL (default), plots are drawn on the current graphics device. Ensure that the file extension matches the desired format (e.g., ".pdf" for PDF output or ".png" for PNG output or .tiff for TIFF output).
ellipse	Logical. Whether to draw a 95\ figures. Default is FALSE.
bg	Logical. Whether to draw the prediction background in the figures. Default is FALSE.
conf_mat	Logical. Whether to print the confusion matrix for the classifications. Default is FALSE.
var_num	Numeric. The number of variables to be used in the PLS-DA model.
cv_opt	Character. Option for cross-validation method: either "loocv" or "Mfold". Default is NULL.
fold_num	Numeric. The number of folds to use if cv_opt is "Mfold". Default is 5.
scale	Character string specifying a transformation to apply to the numeric predictor columns prior to model fitting. Options are "none", "log2", "log10", "zscore", or "custom". When "custom" is selected a user defined function must be supplied via custom_fn. Defaults to "none".
custom_fn	A custom transformation function used when scale = "custom". Ignored otherwise. It should take a numeric vector and return a numeric vector of the same length.
tune	Logical. If TRUE, performs tuning of ncomp and keepX via cross-validation. Default is FALSE.
tune_folds	Integer. Number of folds in cross-validation when tuning. Default is 5.
comp_num	Numeric. The number of components to calculate in the sPLS-DA model. Default is 2.
pch_values	A vector of integers specifying the plotting characters (pch values) to be used in the plots.
style	Character. If set to "3D" or "3d" and comp_num equals 3, a 3D plot is generated using the plot3D package. Default is NULL.

roc	Logical. Whether to compute and plot the ROC curve for the model. Default is FALSE.
verbose	A logical value indicating whether to print additional informational output to the console. When TRUE, the function will display progress messages, and intermediate results when FALSE (the default), it runs quietly.
seed	An integer specifying the seed for reproducibility (default is 123).

Details

When verbose is set to TRUE, additional information about the analysis and confusion matrices are printed to the console. These can be suppressed by keeping verbose = FALSE.

Value

Plots consisting of the classification figures, component figures with Variable of Importance in Projection (VIP) scores, and classifications based on VIP scores greater than 1. ROC curves and confusion matrices are also produced if requested.

Author(s)

Xiaohua Douglas Zhang and Shubh Saraswat

References

Lê Cao, K.-A., Boitard, S. and Besse, P. (2011). Sparse PLS Discriminant Analysis: biologically relevant feature selection and graphical displays for multiclass problems. *BMC Bioinformatics* **12**:253.

Examples

```
# Loading Sample Data
data_df <- ExampleData1[,-c(3)]
data_df <- dplyr::filter(data_df, Group != "ND", Treatment != "Unstimulated")

cyt_splsda(data_df, output_file = NULL,
colors = c("black", "purple"), bg = FALSE, scale = "log2",
conf_mat = FALSE, var_num = 25, cv_opt = NULL, comp_num = 2,
pch_values = c(16, 4), style = NULL, ellipse = TRUE,
group_col = "Group", group_col2 = "Treatment", roc = FALSE, verbose = FALSE)
```

---

cyt_ttest *Two Sample T-test Comparisons.* **[Deprecated]**

Description

This function performs pairwise comparisons between two groups for each combination of a categorical predictor (with exactly two levels) and a continuous outcome variable. It first converts any character variables in `data` to factors and, if specified, applies a log2 transformation to the continuous variables. Depending on the value of `scale`, the function conducts either a two-sample t-test (if `scale = "log2"`) or a Mann-Whitney U test (if `scale` is `NULL`). The resulting p-values are printed and returned.

Usage

```
cyt_ttest(data, scale = NULL, verbose = TRUE, format_output = FALSE)
```

Arguments

`data`	A matrix or data frame containing continuous and categorical variables.
`scale`	A character specifying a transformation for continuous variables. Options are `NULL` (default) and `"log2"`. When `scale = "log2"`, a log2 transformation is applied and a two-sample t-test is used; when `scale` is `NULL`, a Mann-Whitney U test is performed.
`verbose`	A logical indicating whether to print the p-values of the statistical tests. Default is `TRUE`.
`format_output`	Logical. If TRUE, returns the results as a tidy data frame. Default is `FALSE`.

Value

If `format_output` is FALSE, returns a list of p-values (named by Outcome and Categorical variable). If TRUE, returns a data frame in a tidy format.

Author(s)

Shubh Saraswat

Examples

```
data_df <- ExampleData1[, -c(3)]
data_df <- dplyr::filter(data_df, Group != "ND", Treatment != "Unstimulated")
# Test example
cyt_ttest(
 data_df[, c(1:2, 5:6)],
 scale = "log2",
 verbose = TRUE,
 format_output = TRUE
)
```

`cyt_univariate` *Pairwise Univariate Tests Between Two Groups*

Description

`cyt_univariate` supports additional scaling options and explicit choice of statistical test. For each categorical predictor with exactly two levels and each numeric outcome, a two-sample t-test or Wilcoxon rank–sum test is performed. Results are returned either as a list of test objects or, if `format_output = TRUE`, as a tidy data frame with one row per comparison.

Usage

```
cyt_univariate(
  data,
  scale = NULL,
  method = c("auto", "ttest", "wilcox"),
  verbose = TRUE,
  format_output = FALSE,
  custom_fn = NULL,
  p_adjust_method = NULL
)
```

Arguments

`data`	A data frame or matrix containing both categorical and numeric variables.
`scale`	A character specifying a transformation to apply to numeric variables prior to testing. Choices are `NULL` (no transformation), "log2", "log10", "zscore", or "custom". When set to "custom", supply a function via `custom_fn`.
`method`	Character specifying the test to perform. Use "auto" (default) to select between t-test and Wilcoxon based on Shapiro–Wilk normality tests for each outcome; "ttest" to always use Student's t-test; or "wilcox" to always use the Wilcoxon rank–sum test.
`verbose`	Logical indicating whether to return the results. Provided for backward compatibility but has no effect on printing.
`format_output`	Logical. If `TRUE`, returns the results as a tidy data frame; if `FALSE` (default), returns a list of test objects similar to the original function.
`custom_fn`	A function to apply when `scale = "custom"`.
`p_adjust_method`	Character or `NULL`. Method passed to `p.adjust()` for correcting p-values across all comparisons (e.g., `"BH"` for Benjamini-Hochberg). If `NULL` (default) no adjustment is performed.

Value

If `format_output = FALSE`, a named list of test objects keyed by "Outcome_Categorical". If `format_output = TRUE`, a data frame with columns `Outcome`, `Categorical`, `Comparison`, `Test`, `Estimate`, `Statistic`, and `P_value`.

Author(s)

Shubh Saraswat

Examples

```
data_df <- ExampleData1[, -c(3)]
data_df <- dplyr::filter(data_df, Group != "ND", Treatment != "Unstimulated")
cyt_univariate(data_df[, c(1:2, 5:6)], scale = "log2",
               method = "auto", format_output = TRUE)
```

---

cyt_univariate_multi *Univariate Tests for Multi-Level Categorical Predictors*

Description

`cyt_univariate_multi`provides univariate statistical testing for categorical predictors with more than two levels. For each categorical predictor and numeric outcome pair, a global test is performed, followed by pairwise comparisons if the global test is significant. Users may choose between two methods, classical ANOVA with Tukey's Honest Significant Difference (HSD) or a non-parametric Kruskal–Wallis test followed by pairwise Wilcoxon rank–sum tests. The return format can either be a list of adjusted p-values for each outcome–predictor pair or, if `format_output = TRUE`, a tidy data frame summarizing all pairwise comparisons.

Usage

```
cyt_univariate_multi(
  data,
  method = c("anova", "kruskal"),
  cat_vars = NULL,
  cont_vars = NULL,
  p_adjust_method = "BH",
  format_output = FALSE
)
```

Arguments

`data`	A data frame or matrix containing both categorical and continuous variables. Character columns will be converted to factors.
`method`	Character specifying the type of global test to perform. Use "anova" (default) for one-way ANOVA with Tukey HSD or "kruskal" for Kruskal–Wallis with pairwise Wilcoxon tests.
`cat_vars`	Optional character vector of predictor column names. When `NULL`, all factor or character columns in `data` are used.
`cont_vars`	Optional character vector of numeric outcome variable names. When `NULL`, all numeric columns in `data` are used.

p_adjust_method
: Character string specifying the method for p-value adjustment across pairwise comparisons. Passed to p.adjust. Default is "BH".

format_output
: Logical. If TRUE, returns a tidy data frame; otherwise (default) returns a list of numeric vectors keyed by "Outcome_Categorical". Each numeric vector contains adjusted p-values for the pairwise comparisons.

Value

Either a list (if format_output = FALSE) or a data frame (if format_output = TRUE).

Author(s)

Shubh Saraswat

Examples

```
data("ExampleData1")
cyt_univariate_multi(ExampleData1[, c(1:2, 5:6)], method = "kruskal",
                     format_output = TRUE)
```

---

cyt_violin *Violin Plots for Continuous Variables with Optional Grouping*

Description

cyt_violin produces violin plots for each numeric variable in data, optionally grouped by one or more categorical variables. When grouping is not specified, the function behaves similarly to cyt_bp but uses violins instead of boxplots and supports pagination via the bin_size argument. When grouping is provided, a separate violin is drawn for each level (or interaction of levels) of the grouping variables. Users may optionally overlay boxplots within each violin to visualize the median and interquartile range.

Usage

```
cyt_violin(
  data,
  output_file = NULL,
  group_by = NULL,
  bin_size = 25,
  y_lim = NULL,
  scale = c("none", "log2", "log10", "zscore", "custom"),
  custom_fn = NULL,
  boxplot_overlay = FALSE
)
```

Arguments

data	A matrix or data frame containing numeric and categorical variables.
output_file	Optional string specifying the name of the file to be created. When NULL (default), plots are drawn on the current graphics device. Ensure that the file extension matches the desired format (e.g., ".pdf" for PDF output or ".png" for PNG output or .tiff for TIFF output).
group_by	Optional character vector specifying one or more columns to use for grouping. If NULL (default) no grouping is applied.
bin_size	Integer. Maximum number of violins per page when grouping is not used. Default is 25, mirroring cyt_bp.
y_lim	Optional numeric vector giving y-axis limits for the plots. Applies to all plots.
scale	Character specifying a transformation for numeric variables. Accepts "none", "log2", "log10", "zscore", or "custom". When "custom", supply a function via custom_fn.
custom_fn	A user supplied function to transform numeric columns when scale = "custom".
boxplot_overlay	Logical. When TRUE, a narrow boxplot is drawn inside each violin to summarize the median and quartiles. Default is FALSE.

Value

Invisibly returns a list of ggplot objects. When output_file is provided, plots are written to the PDF file.

Author(s)

Shubh Saraswat

Examples

```
# Violin plots without grouping
cyt_violin(ExampleData1[, -c(1:3)], output_file = NULL, scale = "zscore")
# Violin plots grouped by Group with boxplot overlay
cyt_violin(ExampleData1[, -c(3,5:28)], group_by = "Group",
                        boxplot_overlay = TRUE, scale = "log2")
```

---

cyt_volc *Volcano plot*

Description

This function subsets the numeric columns from the input data and compares them based on a selected grouping column. It computes the fold changes (as the ratio of means) and associated p-values for each numeric variable between two groups. The results are log2-transformed (for fold change) and -log10-transformed (for p-values) to generate a volcano plot. Additionally, there is a choice between t-tests and Wilcoxon rank-sum tests and adjusting p-values for multiple comparisons.

Usage

```
cyt_volc(
  data,
  group_col,
  cond1 = NULL,
  cond2 = NULL,
  fold_change_thresh = 2,
  p_value_thresh = 0.05,
  top_labels = 10,
  method = c("ttest", "wilcox"),
  p_adjust_method = NULL,
  add_effect = FALSE,
  verbose = FALSE
)
```

Arguments

data	A data frame containing numeric variables and a grouping column.
group_col	Character. Name of the grouping column.
cond1	Character string specifying the level of group_col for the first condition to compare.
cond2	Character strings specifying the second level of group_col to compare with cond1.
fold_change_thresh	Numeric. Threshold for absolute fold change (in original scale). Default is 2.
p_value_thresh	Numeric. Threshold for the p-value (raw or adjusted). Default is 0.05.
top_labels	Integer. Number of top points to label in each plot. Default is 10.
method	Character. Statistical test to use. "ttest" (default) uses two-sample t-tests; "wilcox" uses Wilcoxon rank-sum tests.
p_adjust_method	Character or NULL. Method to adjust p-values across variables within each comparison (e.g., "BH"). If NULL (default) no adjustment is performed. See p.adjust for details.
add_effect	Logical. If TRUE, effect sizes are computed and returned in the results (Cohen's d for t-tests; rank-biserial for Wilcoxon). Default is FALSE.
verbose	Logical. If TRUE, prints the data frame used for the final comparison without the label column. Default is FALSE.

Value

A list of ggplot objects (one per comparison). Each plot visualizes log2 fold change on the x-axis and –log10 of the (adjusted) p-value on the y-axis. The underlying data used to construct the final plot are printed when verbose = TRUE.

Note

If cond1 and cond2 are not provided, the function automatically generates all possible pairwise combinations of groups from the specified group_col for comparisons.

Author(s)

Xiaohua Douglas Zhang and Shubh Saraswat

Examples

```
# Loading data
data_df <- ExampleData1[,-c(2:3)]

cyt_volc(data_df, "Group", cond1 = "T2D", cond2 = "ND", fold_change_thresh = 2.0, top_labels= 15)
```

---

cyt_xgb *Run XGBoost Classification on Cytokine Data.*

Description

This function trains and evaluates an XGBoost classification model on cytokine data. It allows for hyperparameter tuning, cross-validation, and visualizes feature importance.

Usage

```
cyt_xgb(
  data,
  group_col,
  train_fraction = 0.7,
  nrounds = 500,
  max_depth = 6,
  learning_rate = 0.1,
  nfold = 5,
  cv = FALSE,
  objective = "multi:softprob",
  early_stopping_rounds = NULL,
  eval_metric = "mlogloss",
  min_split_loss = 0,
  colsample_bytree = 1,
  subsample = 1,
  min_child_weight = 1,
  top_n_features = 10,
  verbose = 0,
  plot_roc = FALSE,
  print_results = FALSE,
  seed = 123,
  scale = c("none", "log2", "log10", "zscore", "custom"),
  custom_fn = NULL
)
```

Arguments

data	A data frame containing numeric predictor variables and one grouping column. Non-numeric predictor columns will be coerced to numeric if possible.
group_col	Character string naming the column that contains the class labels (target variable). Required.
train_fraction	Numeric between 0 and 1 specifying the proportion of samples used for model training. The remainder is reserved for testing. Default is 0.7.
nrounds	Integer specifying the number of boosting rounds. Default is 500.
max_depth	Integer specifying the maximum depth of each tree. Default is 6.
learning_rate	Numeric specifying the learning rate. Default is 0.1.
nfold	Integer specifying the number of folds for cross-validation when cv = TRUE. Default is 5.
cv	Logical indicating whether to perform cross-validation using xgb.cv. Default is FALSE.
objective	Character string specifying the objective function. For multi-class classification use "multi:softprob"; for binary classification use "binary:logistic". Default is "multi:softprob".
early_stopping_rounds	Integer specifying the number of rounds without improvement to trigger early stopping. Default is NULL (no early stopping).
eval_metric	Character specifying the evaluation metric used during training. Default is "mlogloss".
min_split_loss	Numeric specifying the minimum loss reduction required to make a further partition. Default is 0.
colsample_bytree	Numeric specifying the subsample ratio of columns when constructing each tree. Default is 1.
subsample	Numeric specifying the subsample ratio of the training instances. Default is 1.
min_child_weight	Numeric specifying the minimum sum of instance weight needed in a child. Default is 1.
top_n_features	Integer specifying the number of top features to display in the importance plot. Default is 10.
verbose	Integer (0, 1 or 2) controlling the verbosity of xgb.train. Default is 0. Larger values print more information.
plot_roc	Logical indicating whether to plot the ROC curve and compute the AUC for binary classification. Default is FALSE.
print_results	Logical. If TRUE, prints the confusion matrix, top features and cross-validation metrics to the console. Default is FALSE.
seed	Optional integer seed for reproducibility. Default is 123.
scale	Character string specifying a transformation to apply to numeric predictors prior to model fitting. Possible values are "none", "log2", "log10", "zscore" or "custom". When set to "custom" a user defined function must be supplied via custom_fn. Defaults to "none".

custom_fn A custom transformation function used when scale = "custom". Ignored otherwise. Should take a numeric vector and return a numeric vector of the same length.

Value

An invisible list with elements: model (the trained xgboost object), confusion_matrix (test set confusion matrix), importance (variable importance matrix), class_mapping (mapping from class names to numeric labels), cv_results (cross-validation results when cv=TRUE), importance_plot (a ggplot2 object of the top feature importance), and roc_plot (a ROC curve for binary classification when plot_roc=TRUE). All plots are printed automatically.

Examples

```
data_df0 <- ExampleData1
data_df <- data.frame(data_df0[, 1:3], log2(data_df0[, -c(1:3)]))
data_df <- data_df[, -c(2:3)]
data_df <- dplyr::filter(data_df, Group != "ND")
cyt_xgb(
 data = data_df,
 group_col = "Group",
 nrounds = 250,
 max_depth = 4,
 min_split_loss = 0,
 learning_rate = 0.05,
 nfold = 5,
 cv = FALSE,
 objective = "multi:softprob",
 eval_metric = "auc",
 plot_roc = TRUE,
 print_results = FALSE)
```

ExampleData1 *Example Cytokine Profiling Data 1.*

Description

Contains observed concentrations of cytokines and their respective treatment and groups, derived from:

Usage

```
ExampleData1
```

Format

A data frame with 297 rows and 29 columns:

Group Group assigned to the subjects.

Treatment Treatment received by subjects.

Time Time point of the measurement.

IL-17F Observed concentration of IL-17F cytokine.

GM-CSF Observed concentration of GM-CSF cytokine.

IFN-G Observed concentration of IFN-G cytokine.

IL-10 Observed concentration of IL-10 cytokine.

CCL-20/MIP-3A Observed concentration of CCL-20/MIP-3A cytokine.

IL-12/P70 Observed concentration of IL-12/P70 cytokine.

IL-13 Observed concentration of IL-13 cytokine.

IL-15 Observed concentration of IL-15 cytokine.

IL-17A Observed concentration of IL-17A cytokine.

IL-22 Observed concentration of IL-22 cytokine.

IL-9 Observed concentration of IL-9 cytokine.

IL-1B Observed concentration of IL-1B cytokine.

IL-33 Observed concentration of IL-33 cytokine.

IL-2 Observed concentration of IL-2 cytokine.

IL-21 Observed concentration of IL-21 cytokine.

IL-4 Observed concentration of IL-4 cytokine.

IL-23 Observed concentration of IL-23 cytokine.

IL-5 Observed concentration of IL-5 cytokine.

IL-6 Observed concentration of IL-6 cytokine.

IL-17E/IL-25 Observed concentration of IL-17E/IL-25 cytokine.

IL-27 Observed concentration of IL-27 cytokine.

IL-31 Observed concentration of IL-31 cytokine.

TNF-A Observed concentration of TNF-A cytokine.

TNF-B Observed concentration of TNF-B cytokine.

IL-28A Observed concentration of IL-28A cytokine.

Source

Example data compiled for cytokine profiling.

References

Pugh GH, Fouladvand S, SantaCruz-Calvo S, Agrawal M, Zhang XD, Chen J, Kern PA, Nikolajczyk BS. T cells dominate peripheral inflammation in a cross-sectional analysis of obesity-associated diabetes. *Obesity (Silver Spring)*. 2022;30(10): 1983–1994. doi:10.1002/oby.23528.

Examples

```
data(ExampleData1)
```

`ExampleData2` *Example Cytokine Profiling Data 2.*

Description

Contains observed concentrations of cytokines and their respective treatment and groups, derived from:

Usage

```
ExampleData2
```

Format

A data frame with 66 rows and 20 columns:

Stimulation Stimulation assigned to the subjects.

Group Group assigned to the subjects.

IL-17F Observed concentration of IL-17F cytokine.

GM-CSF Observed concentration of GM-CSF cytokine.

IFN-G Observed concentration of IFN-G cytokine.

IL-10 Observed concentration of IL-10 cytokine.

CCL-20 Observed concentration of CCL-20 cytokine.

IL-12 Observed concentration of IL-12 cytokine.

IL-13 Observed concentration of IL-13 cytokine.

IL-17A Observed concentration of IL-17A cytokine.

IL-22 Observed concentration of IL-22 cytokine.

IL-9 Observed concentration of IL-9 cytokine.

IL-1B Observed concentration of IL-1B cytokine.

IL-2 Observed concentration of IL-2 cytokine.

IL-21 Observed concentration of IL-21 cytokine.

IL-4 Observed concentration of IL-4 cytokine.

IL-5 Observed concentration of IL-5 cytokine.

IL-6 Observed concentration of IL-6 cytokine.

TNF-A Observed concentration of TNF-A cytokine.

TNF-B Observed concentration of TNF-B cytokine.

Source

Example data compiled for cytokine profiling.

References

SantaCruz-Calvo S, Saraswat S, Hasturk H, Dawson DR, Zhang XD, Nikolajczyk BS. Periodontitis and Diabetes Differentially Affect Inflammation in Obesity. *J Dent Res.* 2024;103(12):1313-1322. doi:10.1177/00220345241280743

Examples

```
data(ExampleData2)
```

`ExampleData3` *Example Cytokine Profiling Data 3.*

Description

Contains observed concentrations of cytokines and their respective treatment and groups, derived from:

Usage

```
ExampleData3
```

Format

A data frame with 64 rows and 14 columns:

Stimulation Stimulation assigned to the subjects.

Group Group assigned to the subjects.

GM-CSF Observed concentration of GM-CSF cytokine.

IFN-G Observed concentration of IFN-G cytokine.

IL-10 Observed concentration of IL-10 cytokine.

CCL-20/MIP-3A Observed concentration of CCL-20/MIP-3A cytokine.

IL-12/P70 Observed concentration of IL-12/P70 cytokine.

IL-13 Observed concentration of IL-13 cytokine.

IL-15 Observed concentration of IL-15 cytokine.

IL-9 Observed concentration of IL-9 cytokine.

IL-1B Observed concentration of IL-1B cytokine.

IL-21 Observed concentration of IL-21 cytokine.

IL-6 Observed concentration of IL-6 cytokine.

TNF-A Observed concentration of TNF-A cytokine.

Source

Example data compiled for cytokine profiling.

References

SantaCruz-Calvo S, Saraswat S, Hasturk H, Dawson DR, Zhang XD, Nikolajczyk BS. Periodontitis and Diabetes Differentially Affect Inflammation in Obesity. *J Dent Res.* 2024;103(12):1313-1322. doi:10.1177/00220345241280743

Examples

```
data(ExampleData3)
```

`ExampleData4` *Example Cytokine Profiling Data 4.*

Description

Contains observed concentrations of cytokines and their respective treatment and groups, derived from:

Usage

```
ExampleData4
```

Format

A data frame with 64 rows and 14 columns:

Group Group assigned to the subjects.

Treatment Treatment received by subjects.

IL-17F Observed concentration of IL-17F cytokine.

GM-CSF Observed concentration of GM-CSF cytokine.

IFN-G Observed concentration of IFN-G cytokine.

IL-10 Observed concentration of IL-10 cytokine.

CCL-20 Observed concentration of CCL-20 cytokine.

IL-12 Observed concentration of IL-12 cytokine.

IL-13 Observed concentration of IL-13 cytokine.

IL-17A Observed concentration of IL-17A cytokine.

IL-22 Observed concentration of IL-22 cytokine.

IL-9 Observed concentration of IL-9 cytokine.

IL-2 Observed concentration of IL-2 cytokine.

IL-21 Observed concentration of IL-21 cytokine.

IL-4 Observed concentration of IL-4 cytokine.

IL-23 Observed concentration of IL-23 cytokine.

IL-5 Observed concentration of IL-5 cytokine.

IL-6 Observed concentration of IL-6 cytokine.

TNF-A Observed concentration of TNF-A cytokine.

TNF-B Observed concentration of TNF-B cytokine.

Source

Example data compiled for cytokine profiling.

References

SantaCruz-Calvo, S., Saraswat, S., Kalantar, G. H., Zukowski, E., Marszalkowski, H., Javidan, A., Gholamrezaeinejad, F., Bharath, L. P., Kern, P. A., Zhang, X. D., & Nikolajczyk, B. S. (2024). A unique inflammaging profile generated by T cells from people with obesity is metformin resistant. *GeroScience*, 10.1007/s11357-024-01441-4. Advance online publication. https://doi.org/10.1007/s11357-024-01441-4

Examples

```
data(ExampleData4)
```

`ExampleData5` *Example Cytokine Profiling Data 5.*

Description

Contains observed concentrations of cytokines and their respective treatment and groups, derived from:

Usage

```
ExampleData5
```

Format

A data frame with 297 rows and 29 columns:

Group Group assigned to the subjects.

Treatment Treatment received by subjects.

Batch Batch number corresponding to the sample.

Time Time point of the measurement.

IL-17F Observed concentration of IL-17F cytokine.

GM-CSF Observed concentration of GM-CSF cytokine.

IFN-G Observed concentration of IFN-G cytokine.

IL-10 Observed concentration of IL-10 cytokine.

CCL-20/MIP-3A Observed concentration of CCL.20.MIP.3A cytokine.

IL-12/P70 Observed concentration of IL-12/P70 cytokine.

IL-13 Observed concentration of IL-13 cytokine.

IL-15 Observed concentration of IL-15 cytokine.

IL-17A Observed concentration of IL-17A cytokine.

IL-22 Observed concentration of IL-22 cytokine.

IL-9 Observed concentration of IL-9 cytokine.

IL-1B Observed concentration of IL-1B cytokine.

IL-33 Observed concentration of IL-33 cytokine.

IL-2 Observed concentration of IL-2 cytokine.

IL-21 Observed concentration of IL-21 cytokine.

IL-4 Observed concentration of IL-4 cytokine.

IL-23 Observed concentration of IL-23 cytokine.

IL-5 Observed concentration of IL-5 cytokine.

IL-6 Observed concentration of IL-6 cytokine.

IL-17E/IL-25 Observed concentration of IL-17E/IL-25 cytokine.

IL-27 Observed concentration of IL-27 cytokine.

IL-31 Observed concentration of IL-31 cytokine.

TNF-A Observed concentration of TNF-A cytokine.

TNF-B Observed concentration of TNF-B cytokine.

IL-28A Observed concentration of IL-28A cytokine.

Note

The ExampleData5 dataset is the same data as ExampleData1 but with a new column "Batch" added to indicate the batch number corresponding to each sample. The "Batch" column was randomly generated to simulate batch effects in the data.

Source

Example data compiled for cytokine profiling.

References

Pugh GH, Fouladvand S, SantaCruz-Calvo S, Agrawal M, Zhang XD, Chen J, Kern PA, Nikolajczyk BS. T cells dominate peripheral inflammation in a cross-sectional analysis of obesity-associated diabetes. *Obesity (Silver Spring)*. 2022;30(10): 1983–1994. doi:10.1002/oby.23528.

Examples

```
data(ExampleData5)
```

Index

∗ **datasets**
ExampleData1, 32
ExampleData2, 34
ExampleData3, 35
ExampleData4, 36
ExampleData5, 37

adjust_p, 3
apply_scale, 3

cyt_anova, 4
cyt_bp, 5
cyt_bp2, 6
cyt_dualflashplot, 7
cyt_errbp, 8
cyt_export, 10
cyt_heatmap, 11
cyt_mint_splsda, 13
cyt_pca, 15
cyt_rf, 17
cyt_skku, 19
cyt_splsda, 21
cyt_ttest, 23
cyt_univariate, 25
cyt_univariate_multi, 26
cyt_violin, 27
cyt_volc, 28
cyt_xgb, 30

ExampleData1, 32
ExampleData2, 34
ExampleData3, 35
ExampleData4, 36
ExampleData5, 37

p.adjust, *29*