

Package: combiroc (via r-universe)

August 28, 2024

Title Selection and Ranking of Omics Biomarkers Combinations Made Easy

Date 2023-07-04

Version 0.3.4

Language en-US

Description Provides functions and a workflow to easily and powerfully calculating specificity, sensitivity and ROC curves of biomarkers combinations. Allows to rank and select multi-markers signatures as well as to find the best performing sub-signatures, now also from single-cell RNA-seq datasets. The method used was first published as a Shiny app and described in Mazzara et al. (2017) <[doi:10.1038/srep45477](https://doi.org/10.1038/srep45477)> and further described in Bombaci & Rossi (2019) <[doi:10.1007/978-1-4939-9164-8_16](https://doi.org/10.1007/978-1-4939-9164-8_16)>, and widely expanded as a package as presented in the bioRxiv pre print Ferrari et al. <[doi:10.1101/2022.01.17.476603](https://doi.org/10.1101/2022.01.17.476603)>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Depends R (>= 3.5.0)

Imports tidy, dplyr, ggplot2, gtools, pROC, stringr, stats, utils, moments

Suggests testthat (>= 3.0.0), knitr, markdown, rmarkdown, httr, Seurat

VignetteBuilder knitr

URL <https://doi.org/10.1101/2022.01.17.476603>,
<https://github.com/ingmbioinfo/combiroc>

BugReports <https://github.com/ingmbioinfo/combiroc/issues>

Config/testthat/edition 3

Repository <https://ingmbioinfo.r-universe.dev>

RemoteUrl <https://github.com/ingmbioinfo/combiroc>

RemoteRef HEAD

RemoteSha ebb7c2eb38c086b6f2187251532a98035e4d28c2

Contents

combi	2
combiroc_long	3
combi_score	4
combs_with	6
demo_data	6
demo_unclassified_data	7
load_data	8
markers_distribution	9
ranked_combs	11
roc_reports	12
seurat_to_combiroc	14
show_markers	15
single_markers_statistics	16

Index **17**

combi	<i>Compute combinations.</i>
-------	------------------------------

Description

A function that computes the marker combinations and counts their corresponding positive samples for each class (once thresholds are selected).

Usage

```
combi(data, signalthr = 0, combithr = 1, max_length = NULL, case_class)
```

Arguments

- | | |
|-----------|---|
| data | a data.frame returned by <code>load_data()</code> . |
| signalthr | a numeric that specifies the value above which a marker expression is considered positive in a given sample. Since the target of the analysis is the identification of marker combinations capable to correctly classify samples, the user should choose a signalthr that: <ul style="list-style-type: none"> • Positively selects most samples belonging to the case class, which must be above signalthr. • Negatively selects most control samples, which must be below signalthr. |

combithr	a numeric that specifies the necessary number of positively expressed markers (\geq signalthr), in a given combination, to consider that combination positively expressed in a sample.
max_length	an integer that specifies the max combination length that is allowed
case_class	a character that specifies which of the two classes of the dataset is the case class

Details

This function counts how many samples are 'positive' for each combination. A sample, to be considered positive for a given combination, must have a value higher than a given signal threshold (signalthr) for at least a given number of markers composing that combination (combithr).

Value

a data.frame containing how many samples of each class are "positive" for each combination, sensitivity and specificity.

Examples

```
## Not run:
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)

# To compute the marker combinations and count their corresponding positive samples for each class.

combs <- combi(data= demo_data, signalthr=450, combithr=1, case_class='A')
# count as positive the samples with value  $\geq$  450 for at least 1 marker in the combination

## End(Not run)
```

combiroc_long	<i>Reshape combiroc data in long format.</i>
---------------	--

Description

A function that simply wraps `dyplr::pivot_longer()` to reshape data in long format without taking into account NAs.

Usage

```
combiroc_long(data)
```

Arguments

data a data.frame returned by `load_data()`.

Details

This function returns the data in long format (with 'Markers' and 'Values' columns)

Value

a data.frame in long format

Examples

```
## Not run:
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)

# To reshape demo_data in long format

demo_data_long <- combiroc_long(data = demo_data)

## End(Not run)
```

combi_score	<i>Compute the combi score using glm(link='binomial') models and optionally classifies the samples.</i>
-------------	---

Description

A function that applies the previously calculated models to a dataset to compute combi scores and optionally classify the samples.

Usage

```
combi_score(
  data,
  Models,
  Metrics,
  Positive_class = 1,
  Negative_class = 0,
  deal_NA = "impute",
  classify = F
)
```

Arguments

data	a data.frame returned by load_data().
Models	a list of glm() objects returned by roc_reports().
Metrics	a list of data.frame objects containing ROC metrics, returned by roc_reports().
Positive_class	a numeric or a character that specifies the label of the samples that will be classified as positives

`Negative_class` a numeric or a character that specifies the label of the samples that will be classified as negatives

`deal_NA` a character that specifies how to treat missing values. With 'impute' NAs of each marker are substituted with the median of that given marker values. With 'remove' the whole observations containing a NA are removed.

`classify` a boolean that specifies if the samples will be classified.

Details

This function can take as input datasets loaded with `load_data()`. They MUST contain all the markers of the combinations used to train the models.

Value

a data.frame containing the combi scores (`classify=F`) or predicted class of each sample (`classify=T`), for each marker/combination in Models

Examples

```
## Not run:
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)
demo_unclassified_data # combiroc built-in unclassified demo data

combs <- combi(data= demo_data, signalthr=450, combithr=1, case_class='A') # compute combinations

reports <- roc_reports(data= demo_data, markers_table= combs,
                      selected_combinations= c(1,11,15),
                      single_markers=c('Marker1', 'Marker2'), case_class='A') # train logistic
                                     # regression models

# To fit the models and retrieve the combi scores (predicted probabilities).
score_data <- combi_score(data= demo_unclassified_data, Models= reports$Models,
                        Metrics= reports$Metrics)

# To classify new samples with logistic regression models.
classified_data <- combi_score(data= demo_unclassified_data,
                             Models= reports$Models, Metrics= reports$Metrics,
                             Positive_class=1, Negative_class=0, classify=TRUE)

classified_data # show samples classified using Logistic regression models

## End(Not run)
```

combs_with	<i>Show combinations with given markers.</i>
------------	--

Description

A function to find all the combinations containing all the markers of interest.

Usage

```
combs_with(markers, markers_table)
```

Arguments

markers a character vector containing one or more markers of interest.
markers_table a data.frame with ranked combination, reporting: SE, SP, number of markers composing the combination and the score (returned by ranked_combs()).

Value

a numeric vector containing the numbers corresponding to the combinations containing all the selected markers.

Examples

```
## Not run:  
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)  
  
combs <- combi(data= demo_data, signalthr=450, combithr=1, case_class='A') # compute combinations  
  
# To show all the combinations with given markers.  
  
combs_with(markers = c('Marker1', 'Marker2') , markers_table = combs)  
  
## End(Not run)
```

demo_data	<i>A demo dataset with proteomics markers</i>
-----------	---

Description

Combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104). A dataset containing signal intensity values of a 5-marker signatures for Autoimmune Hepatitis (AIH). Samples have been clinically diagnosed as “abnormal” (class A) or "normal" (class B).

Usage

```
demo_data
```

Format

A data frame with 170 rows and 7 variables:

Patient.ID the ID of samples

Class the class of the samples: A-Healthy, B-AIH

Marker1 the signal intensity value of Marker1

Marker2 the signal intensity value of Marker2

Marker3 the signal intensity value of Marker3

Marker4 the signal intensity value of Marker4

Marker5 the signal intensity value of Marker5

Examples

```
## Not run:  
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)  
  
## End(Not run)
```

demo_unclassified_data

A demo dataset with unlabeled proteomics markers

Description

Combiroc built-in unclassified demo dataA dataset containing signal intensity values of a 5-marker signatures for Autoimmune Hepatitis (AIH). This dataset has been obtained by randomly picking 20 samples from demo_data without their classification.

Usage

```
demo_unclassified_data
```

Format

A data frame with 20 rows and 7 variables:

Patient.ID the ID of samples

Marker1 the signal intensity value of Marker1

Marker2 the signal intensity value of Marker2

Marker3 the signal intensity value of Marker3

Marker4 the signal intensity value of Marker4

Marker5 the signal intensity value of Marker5

Examples

```
## Not run:
demo_unclassified_data # combiroc built-in unclassified demo data

## End(Not run)
```

load_data	<i>Load CombiROC data.</i>
-----------	----------------------------

Description

A customized read.table() function that checks the conformity of the dataset format, and only if all checks are passed, loads it.

Usage

```
load_data(data, sep = ";", na.strings = "", labelled_data = TRUE)
```

Arguments

data	the name of the file which the data are to be read from.
sep	the field separator character.
na.strings	a character vector of strings which are to be interpreted as NA values.
labelled_data	a boolean that specifies whether the combiroc data to be loaded is labelled (with 'Class' column) or not.

Details

The dataset to be analysed should be in text format, which can be comma, tab or semicolon separated:

- The 1st column must contain patient/sample IDs as characters.
- If dataset is labelled, the 2nd column must contain the class to which each sample belongs.
- The classes must be exactly 2 and they must be written in character format.
- From the 3rd column on (2nd if dataset is unlabelled), the dataset must contain numerical values that represent the signal corresponding to the markers abundance in each sample (marker-related columns).
- Marker-related columns can be called 'Marker1, Marker2, Marker3, ...' or can be called directly with the gene/protein name, but "-" is not allowed in the column name. Only if all the checks are passed, it reorders alphabetically the marker-related columns depending on marker names (necessary for a proper computation of combinations), and it forces "Class" as 2nd column name.

Value

a data frame (data.frame) containing a representation of the data in the file.

Examples

```
## Not run:
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)

# save a data.frame as a csv to be load by combiroc package
file<- tempfile()
write.csv2(demo_data, file = file, row.names = FALSE)

#To load a csv file if correctly formatted

demo_data <- load_data(data = file, sep = ';', na.strings = "")

demo_unclassified_data # combiroc built-in unclassified demo data

# save a data.frame as a csv to be load by combiroc package
file<- tempfile()
write.csv2(demo_unclassified_data, file = file, row.names = FALSE)

# To load an unclassified dataset.

demo_unclassified_data <- load_data(data = file, sep = ';', na.strings = "", labelled_data = FALSE)

## End(Not run)
```

markers_distribution *Show distribution of intensity values for all the markers both singularly and all together.*

Description

A function that takes as input data in long format, and shows how the signal intensity value of markers are distributed.

Usage

```
markers_distribution(  
  data_long,  
  min_SE = 0,  
  min_SP = 0,  
  x_lim = NULL,  
  y_lim = NULL,  
  boxplot_lim = NULL,  
  signalthr_prediction = FALSE,  
  case_class  
)
```

Arguments

<code>data_long</code>	a data.frame in long format returned by <code>combiroc_long()</code>
<code>min_SE</code>	a numeric that specifies the min value of SE that a threshold must have to be included in <code>\$Coord</code> .
<code>min_SP</code>	a numeric that specifies the min value of SP that a threshold must have to be included in <code>\$Coord</code> .
<code>x_lim</code>	a numeric setting the max values of x that will be visualized in the density plot (zoom only, no data loss).
<code>y_lim</code>	a numeric setting the max values of y that will be visualized in the density plot (zoom only, no data loss).
<code>boxplot_lim</code>	a numeric setting the max values of y that will be visualized in the boxplot (zoom only, no data loss).
<code>signalthr_prediction</code>	a boolean that specifies if the density plot will also show the "suggested signal threshold".
<code>case_class</code>	a character that specifies which of the two classes of the dataset is the case class.

Details

This function returns a named list containing the following objects:

- “Density_plot”: a density plot showing the distribution of the signal intensity values for both the classes.
- "Density_summary": a data.frame showing a summary statistics of the distributions.
- “ROC”: a ROC curve showing how many real positive samples would be found positive (SE) and how many real negative samples would be found negative (SP) in function of signal threshold. NB: these SE and SP are refereed to the signal intensity threshold considering all the markers together; it is NOT equal to the SE/SP of a single marker/combination found with `se_sp()`.
- “Coord”: a data.frame that contains the coordinates of the above described “ROC” (threshold, SP and SE) that have at least a min SE (40 by default) and a min SP (80 by default).
- "Boxplot": a boxplot showing the distribution of the signal intensity values of each marker singularly, for both the classes.

In case of lack of a priori known threshold the user can set `signalthr_prediction= TRUE`. In this way the function provides a "suggested signal threshold" that corresponds to the median of the singnal threshold values (in "Coord") at which SE/SP are grater or equal to their set minimal values (`min_SE` and `min_SP`), and it adds this threshold on the "Density_plot" object as a dashed black line. The use of the median allows to pick a threshold whose SE/SP are not too close to the limits (`min_SE` and `min_SP`), but it is recommended to always inspect "Coord" and choose the most appropriate signal threshold by considering SP, SE and Youden index.

Value

a named list containing 'Coord' and 'Density_summary' data.frames, and 'ROC', 'Boxplot' and 'Density_plot' plot objects.

Examples

```
## Not run:
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)

demo_data_long <- combiroc_long(data = demo_data) # long format data

# To visualize the distribution of the expression of each marker.

distributions <- markers_distribution(data_long = demo_data_long,
                                   boxplot_lim = 1500, y_lim = 0.001,
                                   x_lim = 3000 , signalthr_prediction = FALSE,
                                   case_class = 'A', min_SE = 40, min_SP = 80)

distributions$Density_plot # density plot
distributions$Density_summary # summary statistics of density plot
distributions$ROC # ROC showing signal threshold range ensuring min SE and/or SP
distributions$Coord # ROC values
distributions$Boxplot # Boxplot

## End(Not run)
```

ranked_combs

Rank combinations.

Description

A function to rank combinations by a Youden index and select them if they have a min SE and/or SP.

Usage

```
ranked_combs(combo_table, min_SE = 0, min_SP = 0)
```

Arguments

combo_table	a data.frame with SE, SP and number of composing markers for each combination (returned by se_sp()).
min_SE	a numeric that specifies the min value of SE that a combination must have to be filtered-in.
min_SP	a numeric that specifies the min value of SP that a combination must have to be filtered-in.

Details

This function is meant to help the user in finding the best combinations (in the first rows) and allows also (not mandatory) the SE/SP-dependent filtering of combinations.

Value

a named list containing:

- `$table`, a data.frame with ranked combination, reporting: SE, SP, number of markers composing the combination and the score.
- `$bubble_chart`, a dot plot showing the selected 'gold' combinations

Examples

```
## Not run:
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)

combs <- combi(data= demo_data, signalthr=450, combithr=1, case_class='A') # compute combinations
                                     # compute SE and SP
                                     # of each combination

# To rank combinations by Youden index and filter-out the ones with SE < min_SE and SP < min_SP

rc <- ranked_combs(combo_table= combs, min_SE=40, min_SP=80)
rc$table # to visualize the selected gold combinations through a data.frame
rc$bubble_chart # to visualize the selected gold combinations through a data.frame

## End(Not run)
```

roc_reports

Train logistic regression and compute ROC.

Description

A function to compute General Linear Model (binomial) and the corresponding ROC curves for each selected combination.

Usage

```
roc_reports(
  data,
  markers_table,
  selected_combinations = NULL,
  single_markers = NULL,
  case_class,
  deal_NA = "impute"
)
```

Arguments

<code>data</code>	a data.frame returned by <code>load_data()</code> .
<code>markers_table</code>	a data.frame with combinations and corresponding positive samples counts, obtained with <code>combi()</code> .
<code>selected_combinations</code>	a numeric vector that specifies the combinations of interest.
<code>single_markers</code>	a character vector that specifies the single markers of interest.
<code>case_class</code>	a character that specifies which of the two classes of the dataset is the case class.
<code>deal_NA</code>	a character that specifies how to treat missing values. With 'impute' NAs of each marker are substituted with the median of that given marker values in the class that observation belongs to. With 'remove' the whole observations containing a NA are removed'.

Details

This function trains a logistic regression model for each combination and returns a named list containing 3 objects:

- "Plot": a ggplot object with the ROC curves of the selected combinations.
- "Metrics": a data.frame with the metrics of the roc curves (AUC, opt. cutoff, etc ...).
- "Models": the list of models (glm() objects) that have been computed and then used to classify the samples (in which you can find the model equation for each selected combination).

Value

a named list containing 3 objects: "Plot", "Metrics" and "Models".

Examples

```
## Not run:
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)

combs <- combi(data=demo_data, signalthr=450, combithr=1, case_class='A') # compute combinations

# To train logistic regression models on each selected combinations and
# each selected marker, and compute corresponding ROCs.

reports <- roc_reports(data= demo_data, markers_table= combs,
                      selected_combinations= c(1,11,15),
                      single_markers=c('Marker1', 'Marker2'), case_class='A')

reports$Plot # Shows the ROC curves
reports$Metrics # Shows the ROC metrics
reports$Models # show models
reports$reports$Models$`Combination 11` # show model trained with Combination 11

## End(Not run)
```

seurat_to_combiroc *Load CombiROC data.*

Description

A function to extract a combiroc data (both labelled and unlabelled) from a SeuratObject.

Usage

```
seurat_to_combiroc(  
  SeuratObject,  
  gene_list,  
  assay = "RNA",  
  labelled_data = F,  
  case_class = NA,  
  case_label = "case",  
  control_label = "control"  
)
```

Arguments

SeuratObject	Defines S4 classes for single-cell genomic data and associated information.
gene_list	a list of gene names.
assay	a character that specifies the assay of interest.
labelled_data	a boolean that specifies whether the combiroc data to be extracted must be labelled (with 'Class' column) or not.
case_class	a character or a character vector specifying the category/ies to be considered as 'case class'. Required if labelled_data is TRUE.
case_label	a character that will be assigned to the cells belonging to 'case class' category.
control_label	a character that will be assigned to the cells belonging to 'control class' category.

Details

By specifying a gene list (if the genes are in rownames of SeuratObject assay matrix, it subsets the gene expression matrix (@data) and it retrieves their expression values. If a combiroc training dataset (labelled_data=T) is required, one or more categories of Idents(SeuratObject) must be selected as case_class, the others will be merged into control class.

Value

a combiroc data.

Examples

```
## Not run:
demo_seurat# A subset of PBMC3K dataset from Satijia et al. 2015

# list of markers of interest
gene_list<-c('RBP7','CYP1B1','CD14','FCN1','NKG7', 'GNLY')

# to extract the combircoc data from data from pbmc3k.final (case class: Monocytes)
data <- seurat_to_combircoc(demo_seurat, gene_list = gene_list, labelled_data = TRUE,
                           case_class = c('CD14+ Mono', 'FCGR3A+ Mono'),
                           case_label = 'Monocyte', control_label='Other')

head(data)

## End(Not run)
```

show_markers

Show the composition of combinations.

Description

A function to show the composition of combinations of interest.

Usage

```
show_markers(markers_table, selected_combinations)
```

Arguments

markers_table a data.frame with combinations returned by combi().
selected_combinations
a numeric vector that specifies the combinations of interest.

Value

a data.frame containing the selected combinations and their composing markers.

Examples

```
## Not run:
demo_data # combircoc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)

combs <- combi(data= demo_data, signalthr=450, combithr=1, case_class='A') # compute combinations

# To show the composition of combinations of interest.

show_markers(markers_table = combs, selected_combinations = c(1,11))

## End(Not run)
```

`single_markers_statistics`*Provide statistics for each marker.*

Description

A function that computes the statistics and a scatter-plot for each marker.

Usage

```
single_markers_statistics(data_long)
```

Arguments

`data_long` a data.frame in long format returned by `combiroc_long()`.

Details

This function computes the main statistics of the signal values distribution of each marker in both classes. In addition it also shows the values through scatter plots.

Value

a list object containing:

- 'Statistics': a dataframe containing the main statistics for each marker in each class.
- 'Plots': a named list of scatter plots showing signal intensity values.

Examples

```
## Not run:
demo_data # combiroc built-in demo data (proteomics data from Zingaretti et al. 2012 - PMC3518104)

data_long <- combiroc_long(demo_data) # reshape data in long format

sms <- single_markers_statistics(data_long)

sms$Statistics # to visualize the statistics of each single marker
sms$Plots[[1]] # to visualize the scatterplot of the first marker

## End(Not run)
```


Index

* datasets

demo_data, [6](#)

demo_unclassified_data, [7](#)

combi, [2](#)

combi_score, [4](#)

combiroc_long, [3](#)

combs_with, [6](#)

demo_data, [6](#)

demo_unclassified_data, [7](#)

load_data, [8](#)

markers_distribution, [9](#)

ranked_combs, [11](#)

roc_reports, [12](#)

seurat_to_combiroc, [14](#)

show_markers, [15](#)

single_markers_statistics, [16](#)