#' Example function 01

#'

#' A theoretical proteomics dataset composed of 1000 human proteins (UniProt accessionn umbers) and random choosen values will be used to call the clustpro() main function.

#' @return see clustpro() function output

#' @export

#'

clustpro\_example <- function(){}

#' Clustpro main function

#'

#' This function is used to start the clustering and visualisation process.

#' @param matrix a data.frame of values for which the clustering is performed

#' @param method a character string specifying the the clustering method (‘kmeans’,’cmeans’)

#' @param min\_k,max\_k,fixed\_k for all potentional k in the range of min\_k and max\_k the Davies–Bouldin index is calculated. If fixed\_k is set to an integer 1< no Davies-Bouldin test is performed and k is set to the fixed value.

#' @param perform\_clustering logical; if TRUE, data will be clustered

#' @param cluster\_ids NOT IN USE

#' @param rows,cols logical; TRUE for reordering by hierarchical clustering of cluster/columns

#' @param tooltip list of lists; requires ‘id’ list containing identifiers (e.g. rownames). Additional list entries were added to the tooltip in the same order as in the global tooltip list. List with name ‘link’ will added as link to the cells. Website is shown by double click.

#' @param save\_widget NOT IN USE

#' @param color\_legend

#' @param width,height

#' @param export\_dir

#' @param export\_type

#' @param seed

#' @return see clustpro() function output

#' @examples

#' @export

clustpro <- function(){}

#' Shiny bindings for clustpro

#'

#' Output and render functions for using clustpro within Shiny

#' applications and interactive Rmd documents.

#'

#' @param outputId output variable to read from

#' @param width,height Must be a valid CSS unit (like \code{'100\%'},

#' \code{'400px'}, \code{'auto'}) or a number, which will be coerced to a

#' string and have \code{'px'} appended.

#' @param expr An expression that generates a clustpro

#' @param env The environment in which to evaluate \code{expr}.

#' @param quoted Is \code{expr} a quoted expression (with \code{quote()})? This

#' is useful if you want to save an expression in a variable.

#'

#' @name clustpro-shiny

#'

#' @export

clustproOutput <- function(){}

#' Shiny bindings for clustpro

#'

#' Output and render functions for using clustpro within Shiny

#' applications and interactive Rmd documents.

#'

#' @param outputId output variable to read from

#' @param width,height Must be a valid CSS unit (like \code{'100\%'},

#' \code{'400px'}, \code{'auto'}) or a number, which will be coerced to a

#' string and have \code{'px'} appended.

#' @param expr An expression that generates a clustpro

#' @param env The environment in which to evaluate \code{expr}.

#' @param quoted Is \code{expr} a quoted expression (with \code{quote()})? This

#' is useful if you want to save an expression in a variable.

#'

#' @name clustpro-shiny

#'

#' @export

clustproOutput <- function(){}

#' distributions\_histograms

#'

#' .................

#' @param matrix

#' @examples

distributions\_histograms <- function() {}

#' order\_dataframe\_by\_list

#'

#' .................

#' @param x

#' @param list

#' @param col

#' @param reverse

#' @examples

order\_dataframe\_by\_list <- function(){}

#' findk\_cmeans

#'

#' .................

#' @param matrix

#' @param k

#' @param minimalSet

#' @param fp

#' @param seed

#' @examples

findk\_cmeans <- function(){}

#' findk\_kmeans

#'

#' .................

#' @param matrix

#' @param k

#' @param seed

#' @examples

findk\_kmeans <- function(){}

#' Get best k

#'

#' .................

#' @param matrix

#' @param min\_k,max\_k,fixed\_k

#' @param method

#' @param no\_cores

#' @param seed

#' @examples

get\_best\_k <- function(){}

#' Clustering

#'

#' This function allows you to initialize a graphic

#' @param matrix

#' @param min\_k,max\_k,fixed\_k

#' @param method

#' @param no\_cores

#' @param seed

#' @examples

clustering <- function(){}

#' Function to initialize a graphic

#'

#' This function allows you to initialize a graphic

#' @param title

#' @param project

#' @param type

#' @param number

#' @examples

initialize\_graphic <- function(){}

#' Function to get color

#'

#' ..............

#' @param x

#' @param ticks

#' @param colors

#' @examples

get\_color <- function(){}

#' Function to to define the color spectrum for heatmaps

#'

#' This function allows you to define the color spectrum for heatmaps.

#' @param values should be a list which define the breaks of the color space. color\_spect should be a list of color. Keep in mean that there must be 1 more board in the vaules list than color in color\_spect.

#' @param color\_spect

#' @param shift\_factor

#' @examples

#' color\_spectrum()

color\_spectrum <- function(){}

#' Function to set heatmap color

#'

#' This function allows you to define the color spectrum for heatmaps.

#' @param data todo

#' @param color\_list todo

#' @param intervals todo

#' @param auto todo

#' @keywords color spectrum heatmaps

#' @export

#' @examples setHeatmapColors()

setHeatmapColors <- function(){}