hierarchical.py Documentation

author: Samir Farooq
company: University of Rochester Medical Center
team: Rochester Center for Health Informatics
email: samir_farooq@urmc.rochester.edu or martin_zand@urmc.rochester.edu
last revised: 13 June 2018

We noticed that the data structure of the output of scipy.cluster.hierarchy.**dendrogram** was not clearly constructed. The clustering results were therefore difficult to interpret merely due to the choice of the data structure, particularly when the number of clusters were greater than seven (due to repeated color usage). This python script is meant to fix the issue with a more user-friendly data structure.

class `hierarchical.linkage_clustering (Z, thresh=None, classes=None, label=None):`

This class has been designed to be more versatile than scipy.cluster.hierarchy.**dendrogram** and provide more meaningful results to the end-user; which is to use linkage clustering results to create cluster structures based on a provided threshold. Note that running `linkage_clustering` will act to perform the structuring of the clusters, but will not plot anything. For plotting please either first use the method `generate_cluster_colors` if you wish to assign specific color the classes, then/or use the method `plot_dendrogram` (which does not need `generate_cluster_colors` to have already been run to operate).

Parameters:

`Z` : numpy array, shape = [n_rows - 1, 4]

This numpy array should come from the result of performing scipy.cluster.hierarchy.**linkage** on a matrix.

`thresh` : None or float, (default=None)

The specific threshold to make a cut on a hierarchical dendrogram. If `thresh`=None then the threshold is set to the maximum distance between clusters (i.e. a cut is made above any branches; hence only one cluster is identified).

`classes` : numpy array or None, (default=None)

The classes of the rows (if applicable). Generally this would be provided to see how clustering results compare to class label segregation.

`label` : numpy array or None, (default=None)

These are the labels of the rows (e.g. patients, if working with patient data). If `label`=None then any subsequent plots generated from this class structure will not be labeled on the x-axis.

Attributes:

`Clusters` : list

A list of sets, where the sets contain the patients (rows) belonging to the same cluster. Hence the length of `Clusters` is equal to the number of clusters.

`colors` : dictionary

Does not exist as an attribute until either `generate_cluster_colors` is run or `plot_dendrogram` is run. The keys are the cluster numbers (corresponding to the index of list `Clusters`), and the values are color names or RGB color codes.

def_bracket_color : numpy array, shape = [3,]

Does not exist as an attribute until either generate_cluster_colors is run or plot_dendrogram is run. This is the color which the brackets above the cutting threshold will have. While there is no method to change this color, one can change it manually by first running generate_cluster_colors followed by manually changing this attribute to the desired color.

Z : numpy array, shape = [n_rows - 1, 4]

The same Z which was inputted as the parameter.

thresh : float or None

The same thresh which was inputted as the parameter.

classes : list or None

the classes which were inputted.

label : list or None

The labels which were inputted.

Zt : numpy array

This Z with the threshold applied to it.

B : dictionary

Contains a reference to each patient (or row) to which cluster number the patient belongs. The cluster number is the same as the index number of the for the Clusters attribute.

p : integer

The number of patients (rows).

cn : integer

Same as len(Clusters).

S : set

Record keeper, not to be touched by end-user.

C : set

Record keeper, not to be touched by end-user.

Methods:

generate_cluster_colors (colorblind_friendly=True, set_own=False)

If colorblind_friendly=True, then there cannot be any more than 8 clusters. These colors are taken from http://mkweb.bcgsc.ca/colorblind/. If one wishes to set their own colors then they can do so by first setting colorblind_friendly=False, then changing set_own to a dictionary: where the keys are the cluster numbers (corresponding to the index of Clusters), and the values are the color names or RGB color codes. If both inputs are false then we draw equally distributed colors from the hue color-map.

plot_dendrogram (ax=None, title=False, class_color=None, label=False, new_thresh=False, orientation='top', reverse_labelflush=False, color_label=True, yflip=True, xtick_rotate=0)

Plots the dendrogram on a new figure if no ax is provided, otherwise plots the dendrogram on ax. To turn on or off labels on x-axis or to adjust the cut threshold, then one will have to re-run `linkage_clustering` from scratch.

- `ax`: controls the axis to plot the figure on. If ax=None or ax='small', then it will plotted on a small sized figure. If ax='med' or if ax='big' then it will be plotted on a medium or big sized figure respectively.
- `title`: The title of the plot. Set to False if a title is not desired, otherwise type the title as a string.
- `class_color`: The color of the classes (as long as the `classes` attribute is not None. The input should be a dictionary with the keys being the classes and its value being a recognized color code by matplotlib (i.e. rgb or string codes).
- `label`: Whether or not to label the leaves (True or False).
- `new_thresh`: This parameter is meant to control whether or not the user wishes to try a new_threshold in the plotting (i.e. by providing a new float value) - Warning: there are currently unresolved issues in using this parameter, hence we suggest to create a new instance of this class with a new `thresh` and then plotting instead of using this parameter.
- `orientation`: Controls which direction the trunk of the tree should be facing: 'top' (default), 'bottom', 'left', or 'right' are accepted inputs.
- `reverse_labelflush`: Whether or not the reverse the flush direction of the labels (True or False).
- `color_label`: Whether or not to color the labels in accordance with the cluster colors (True or False).
- `yflip`: If `orientation` == 'right' or 'left', then `yflip` controls whether or not to flip the y-axis (True or False).
- `xtick_rotate`: If `orientation` == 'bottom' or 'top' then `xtick_rotate` controls the degree of rotation of the labels (as long as `label`=True), hence the input should be an integer or float corresponding to degree rotation.

`get_plot_boundaries ()`

Returns the boundary coordinates of each class in the plotted dendrogram in the form of a dictionary where the keys are the classes and the value is a list of three values: [minimum, average, maximum]. This of course means `plot` must be run before any boundaries can be returned.

`add_subcomponent (i, cluster)`

This is a back-end method which is not meant for the user to play around with- responsible for structuring the patients together.

`plot_segment (x, bot, top, color)`

This is a sub-routine of `plot_dendrogram` and should not be touched by user.

`check_position (index)`

This is a sub-routine of `plot_dendrogram` and should not be touched by user.

`update_bracket (zi)`

This is a sub-routine of `plot_dendrogram` and should not be touched by user.

def `hierarchical.cluster_gains (Z, classes, plot=False):`

Measures the amount of information that clusters yield in relation to their true classes using entropy based formulas.

`Input:`

`Z` : numpy array, shape = [n_rows - 1, 4]

This numpy array should come from the result of performing scipy.cluster.hierarchy.**linkage** on a matrix.

`classes` : numpy array or list

The list of true class labels of the rows.

`plot` : boolean, (default=False)

Whether or not to plot the information gain of the resulting clusters.

`Output:`

`area` : float

A value between 0-1 which indicates how accurately the clusters match up to the real classes. A value of 0 indicates no match and a value of 1 indicates perfect match.