Package 'flowMagic'

January 24, 2024

Title Automated gating of flow cytometry data

Version 0.99

Description tool to perform the automated gating of flow cytometry data in a user friendly way.

License GPL (>= 2)

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Imports caret

R topics documented:

.densRange . . . 2
add_labels_column . . . 3
assign_events_to_nearest_centroids . . . 3
check_polygons_intersection . . . 4
compute_gates . . . 5
csv_to_dens . . . 5
exports_plots . . . 6
extract_polygon_gates . . . 7
get_centroids . . . 7
get_classes_expr_df . . . 8
get_density_features . . . 8
get_density_scores . . . 9
get_distance_loc_vs_test . . . 9
get_dist_template . . . 10
get_hierarchy_all_pops . . . 11
get_hull_all_gates . . . 11
get_indices_cross_val . . . 12
get_local_train_sets . . . 12
get_paths_training . . . 13
get_paths_training_v2 . . . 14
get_pops_hierarchy_list . . . 14

get_pop_multiclass 15
get_slot_hierarchy_list 15
get_test_sets 16
get_train_data 16
get_weights_density_features 17
import_png_image 18
import_reference_csv 18
import_sample_gated 19
import_test_set 19
import_test_set_csv 20
magicPlot 20
magicPred 21
magicPred_all 22
magicPred_hierarchy 23
magicTrain 23
magicTrain_dt 24
magicTrain_local 25
magicTrain_nb 25
magicTrain_nnet 26
magicTrain_rf 27
name_pop_gating 27
post_process_gates 28
pre_process_manual_binary 29
process_test_data 29
range01 30
subsetting_binary_df 30
update_label_association 31

Index **32**

`.densRange` *.densRange*

Description

function to get the correct coordinates (correct range) based on the density coordinates x and y generated by the density() function.

Usage

```
.densRange(x, y, gate, pos = FALSE)
```

Arguments

`x`	Coordinates of x axis.
`y`	Coordinates of y axis
`gate`	gate threshold.
`pos`	refere to the pos in deGatePlot() function.

Value

list of numbers.

Examples

```
.densRange()
```

`add_labels_column` *add_labels_column*

Description

function to add label association column.

Usage

```
add_labels_column(df, labels_assocation)
```

Arguments

`df`	Dataframe.
`labels_assocation`	Vector of label association.

Value

Dataframe.

Examples

```
add_labels_column()
```

`assign_events_to_nearest_centroids` *assign_events_to_nearest_centroids*

Description

function to assign events to class with nearest centroid.

Usage

```
assign_events_to_nearest_centroids(gated_df, n_cores = 1,
  method_dist = "euclidean", thr_dist = 0.15, include_zero = F,
  remove_centroids = T)
```

Arguments

gated_df	dataframe with labels (third column).
n_cores	Number of cores. Default to 1.
method_dist	Distance method calculation. Default to euclidean.
thr_dist	Distance threshold for centroids calculation. Default to 0.15.
include_zero	Consider centroid of label 0. Default to False.
remove_centroids	Remove centroids too near each other based on thr_dist value.

Value

Dataframe.

Examples

```
assign_events_to_nearest_centroids()
```

---

check_polygons_intersection *extract_polygon_gates*

Description

funcction to check polygons intersection.

Usage

```
check_polygons_intersection(list_df_hull)
```

Arguments

list_df_hull	List of polygons coordinates

Value

float

Examples

```
check_polygons_intersection()
```

compute_gates *compute_gates*

Description

function to assign events based on polygon gates.

Usage

```
compute_gates(gated_df, list_final_polygons_coords, no_classes = F)
```

Arguments

gated_df	dataframe with labels (third column).
list_final_polygons_coords	List of dataframes containing polygon coordinates.
no_classes	Generate third column of labels. Default to False.

Value

Dataframe.

Examples

```
compute_gates()
```

csv_to_dens *csv_to_dens*

Description

function to get density of events (with classes associated if present).

Usage

```
csv_to_dens(df, with_classes = T, n_coord = "df")
```

Arguments

df	Dataframe of marker expression values.
with_classes	Consider classes. Default to True.
n_coord	Grid size. Default to df.

Value

Dataframe.

Examples

```
csv_to_dens()
```

`exports_plots` *exports_plots*

Description

function to generate plots (no hierarchy).

Usage

```
exports_plots(list_gated_data, path_output, n_cores = 1,
  type_plot = "dens", show_legend = T, x_lab = "x", y_lab = "y",
  size_title_x = 23, size_title_y = 23)
```

Arguments

`list_gated_data`	list of dataframes. Each dataframe has 3 columns: marker 1 values, marker 2 values and label column.
`path_output`	Path to the directory where to export the plots.
`n_cores`	Path to directory containing the expression data of the files analyzed.
`type_plot`	the user can choose between density (="dens) or label assignment visualization (="ML")
`show_legend`	If True it shows the legend for the label assignment visualization. Default to True.
`x_lab`	x-axis label.
`y_lab`	y-axis label.
`size_title_x`	Size x axis label.
`size_title_y`	Size y axis label.

Examples

```
exports_plots()
```

extract_polygon_gates *extract_polygon_gates*

Description

function to extract the polygon gates objects based on the convex hull and classes.

Usage

```
extract_polygon_gates(gated_df, concavity_val = 1)
```

Arguments

gated_df	dataframe with labels (third column).
concavity_val	Concavity of polygons. Default to 1.

Value

List of dataframes.

Examples

```
extract_polygon_gates()
```

get_centroids *get_centroids*

Description

function to get centroids for each label

Usage

```
get_centroids(df, low_thr = 0.1, up_thr = 0.9, thr_dist = 0.15,
  include_zero = F, remove_centroids = T)
```

Arguments

df	dataframe with labels (third column).
low_thr	Lower threshold for quantile calculation.
up_thr	Upper threshold for quantile calculation.
thr_dist	Distance threshold for centroids calculation. Default to 0.15.
include_zero	Consider centroid of label 0. Default to False.
remove_centroids	Remove centroids too near each other based on thr_dist value.

Value

Dataframe.

Examples

```
get_centroids()
```

---

get_classes_expr_df *get_classes_expr_df*

Description

function to get classes of original expression df based on density df predictions.

Usage

```
get_classes_expr_df(dens_df, original_df)
```

Arguments

dens_df	Dataframe of density values.
original_df	Original dataframe.

Value

Dataframe.

Examples

```
get_classes_expr_df()
```

---

get_density_features *get_density_features*

Description

function to get density features only given a bivarite density csv.

Usage

```
get_density_features(df_dens, min_height = 0.06)
```

Arguments

df_dens	Dataframe of density estimates for both markers.
min_height	Minimum height of the peaks to consider.

Value

Vector of numbers.

Examples

```
get_density_features()
```

---

get_density_scores *get_density_scores*

Description

function to get scores for distance template calculation.

Usage

```
get_density_scores(df_template, df_test, select_density_features = NULL)
```

Arguments

df_template	Dataframe of template markers expression values.
df_test	Dataframe of test markers expression values.
select_density_features	Select features to use. Default to NULL.

Value

Matrix of numbers.

Examples

```
get_density_scores()
```

---

get_distance_loc_vs_test *get_distance_loc_vs_test*

Description

function to compare the similiraty between current test data and local training set.

Usage

```
get_distance_loc_vs_test(test_df, loc_df, show_plot = "none",
  nboot = 50)
```

Arguments

test_df	Dataframe with bivariate marker expression.
loc_df	Dataframe with bivariate marker expression.
show_plot	show density comparison. Default to none.
nboot	Number of permutations. Default to 50.

Value

List of p-values.

Examples

```
get_distance_loc_vs_test()
```

---

get_dist_template *get_dist_template*

Description

function to get distance between template and test data.

Usage

```
get_dist_template(matrix_scores, dist_method = "euclidean")
```

Arguments

matrix_scores	Matrix of density features generated by get_density_scores function.
dist_method	Type of distance method calculation.

Value

Number.

Examples

```
get_dist_template()
```

get_hierarchy_all_pops *get_hierarchy_all_pops*

Description

function to get the hierarchy of all pops from the sample manually gated (the input gating hierarchy). Based on its output the function magicTrain will perform the training step using the sample manually gated (local training set) and the project discovery data (global training set).

Usage

```
get_hierarchy_all_pops(gh, export_visnet = F, path.output = "None")
```

Arguments

gh	GatingHierarchy.
export_visnet	If true, it export visnetwork object.
path.output	Path to save visnetwork object. Default to None.

Value

List of Dataframes.

Examples

```
get_hierarchy_all_pops()
```

get_hull_all_gates *get_hull_all_gates*

Description

function to get the convex hull of all gates.

Usage

```
get_hull_all_gates(gated_df, concavity_val = 1)
```

Arguments

gated_df	dataframe with labels (third column).
concavity_val	Values of concavity. Default to 1.

Value

List of dataframes.

Examples

```
get_hull_all_gates()
```

---

get_indices_cross_val *get_indices_cross_val*

Description

function to get indices for cross val.

Usage

```
get_indices_cross_val(df_train, n_cores = 1)
```

Arguments

df_train	Dataframe of training features generated by the get_train_data function.
n_cores	Number of cores to use. Default to 1.

Value

List of numbers.

Examples

```
get_indices_cross_val()
```

---

get_local_train_sets *get_local_train_sets*

Description

Based on the hierarchy calculated using the get_hierarchy function, we generate the local training sets gated by the biologists.

Usage

```
get_local_train_sets(gh, hierarchical_tree, info_hierarchy)
```

Arguments

gh	GatingHierarchy.
hierarchical_tree	Dataframe of hierarchy information generated by the get_hierarchy function.
info_hierarchy	List of hierarchy information generated by the get_hierarchy function.

Value

List of Dataframes.

Examples

```
get_local_train_sets()
```

---

get_paths_training *get_paths_training*

Description

function to get paths to train data.

Usage

```
get_paths_training(df_paths, n_paths = 200, seed_n = 40)
```

Arguments

df_paths	Dataframe of paths to training data.
n_paths	Max number of random paths to consider for number of gates.
seed_n	Random seed. Default to 40.

Value

Vector of characters.

Examples

```
get_paths_training()
```

get_paths_training_v2 *get_paths_training_v2*

Description

function to get paths to train data.

Usage

```
get_paths_training_v2(df_paths, n_paths = 200, seed_n = 40)
```

Arguments

df_paths	Dataframe of paths to training data.
n_paths	Max number of random paths to consider for number of gates.
seed_n	Random seed. Default to 40.

Value

Vector of characters.

Examples

```
get_paths_training_v2()
```

get_pops_hierarchy_list *get_pops_hierarchy_list*

Description

function to get info from a list of hierarchical dataset (it reports all the pops for each level in a vector).

Usage

```
get_pops_hierarchy_list(hierarchical_list)
```

Arguments

hierarchical_list	list of populations names for each level.

Value

Vector of characters.

Examples

```
get_pops_hierarchy_list()
```

`get_pop_multiclass` *get_pop_multiclass*

Description

function to get the children of the selected population, useful for multiclass classification. The output is only the pops with same dimensions

Usage

```
get_pop_multiclass(gh, pop)
```

Arguments

gh	GatingHierarchy.
pop	Name of population to get events assignments.

Value

Vector of characters.

Examples

```
get_pop_multiclass()
```

`get_slot_hierarchy_list` *get_slot_hierarchy_list*

Description

function to access a slot of results from the hierarchy list based on the pop selected.

Usage

```
get_slot_hierarchy_list(hierarchical_list, pop_selected)
```

Arguments

hierarchical_list	list of populations names for each level.
pop_selected	Population name.

Value

Element of a list.

Examples

```
get_slot_hierarchy_list()
```

`get_test_sets` *get_test_sets*

Description

Function to obtain the data ready to be gated (validation set or test set). It takes as input a flowSet and generates a list of dataframes with a ML structure for each ungated fcs file. Each dataframe is the expression matrix of the root.

Usage

```
get_test_sets(fs, gh)
```

Arguments

`fs`	flowSet to gate.
`gh`	Gating hierarchy.

Value

List of dataframes.

Examples

```
get_test_sets()
```

`get_train_data` *get_train_data*

Description

function to import test set in csv format.

Usage

```
get_train_data(paths_file = NULL, df_paths = NULL, n_cores = 1,
  prop_down = NULL, remove_class = NULL)
```

Arguments

paths_file	Vector of paths. Each path points toward a single csv file containin training info (labels and bivariate expression).
df_paths	Dataframe containing the paths of file to read. The paths must be in the first column.
n_cores	Number of cores. Default to 1.
prop_down	Proportion of events to consider (dowsampling). Default to NULL.
remove_class	Vector of classes to ignore. Default to NULL.

Value

Dataframe.

Examples

```
get_train_data()
```

---

get_weights_density_features
get_weights_density_features

Description

function to calculate final score based on density features.

Usage

```
get_weights_density_features(df_scores)
```

Arguments

df_scores	Dataframe of distance scores.

Value

Vector of numbers.

Examples

```
get_weights_density_features()
```

import_png_image *import_png_image*

Description

function to correctly import a png image as matrix of pixels.

Usage

```
import_png_image(path_img)
```

Arguments

path_img	path to directory containing csv files to read (third column is ignored).

Value

matrix.

Examples

```
import_png_image()
```

import_reference_csv *import_reference_csv*

Description

function to import plain gold standards data (no hierarchy)

Usage

```
import_reference_csv(path_results, n_cores = 1)
```

Arguments

path_results	path to directory containing the csv files to read (with third column of labels).
n_cores	Number of cores to use. Default to 1.

Value

list of dataframes

Examples

```
import_reference_csv()
```

import_sample_gated *import_sample_gated*

Description

convert a flowWorkspace or a GatingMl file of the train data folder into a gated gh (gh_train)

Usage

```
import_sample_gated(path, type = "gs", group_wsp = NULL,
  report_gs = F)
```

Arguments

path	path to gs or GatingML object.
type	type of object.
group_wsp	Group of wsp to import.
report_gs	report gs?

Value

GatingHierarchy object

Examples

```
import_sample_gated()
```

import_test_set *import_test_set*

Description

read the ungated fcs files into a flowSet. The ungated fcs are assumed to be already cleaned,compensated,and transformed.

Usage

```
import_test_set(path, n_samples = "All", ref_f_n = 1)
```

Arguments

path	path of directory containig the fcs files.
n_samples	Number of samples. Default to All.
ref_f_n	Set reference flowFrame to match channel names. Default to 1(first flowFrame).

Value

flowSet.

Examples

```
import_test_set()
```

`import_test_set_csv` *import_test_set_csv*

Description

function to import test set in csv format.

Usage

```
import_test_set_csv(path_data, n_cores = 1, xy_col = T)
```

Arguments

`path_data`	path to directory containing csv files to read (third column is ignored).
`n_cores`	Number cores. Default to 1.
`xy_col`	Colnames equal to x and y. Default to True.

Value

List of dataframes.

Examples

```
import_test_set_csv()
```

`magicPlot` *magicPlot*

Description

function to generate the scatter plot with colored density of the events.

Usage

```
magicPlot(df, type = "dens", polygons_coords_list = NULL,
  show_legend = T, size_axis_text = 18, size_title_x = 20,
  size_title_y = 20, treat_0_as_gate = F, x_lab = "x", y_lab = "y",
  gates_to_plot = NULL, apply_manual_scale = T, size_points = 1,
  concavity_val = 5)
```

Arguments

df	Dataframe of bivariate markers expression (with labels if gates to plot).
type	Type of plot to generated. "dens"=bivariate density plot. "ML"=events assignments plot
polygons_coords_list	list of gates coordinates. Needed if labels not included in df. Default to NULL.
show_legend	Show legend if type="ML". Default to True.
size_axis_text	Size of axis ticks labels. Default to 18.
size_title_x	Size of x axis label title.
size_title_y	Size of y axis label title.
treat_0_as_gate	Treat 0 label as gate. Defaul to False (0 label is background)
x_lab	Label of x axis.
y_lab	Label off y axis.
gates_to_plot	Select labels to plot.
apply_manual_scale	Apply predifined scale of colors. Default to True.
size_points	Size of points in scatter plot.
concavity_val	Concavity value. Default to 5.

Value

ggplot.

Examples

```
magicPlot()
```

---

magicPred *magicPred*

Description

function to predict on plain test data (no hierarchy)

Usage

```
magicPred(test_data, magic_model, ref_model_info = NULL, n_cores = 1,
  ref_data_train = NULL, prop_down = 1)
```

Arguments

test_data	Dataframe of test data to gate. It has only the two columns of marker expression.
magic_model	Global trained model.
ref_model_info	Template model
n_cores	Number of cores to use. Default to 1.
ref_data_train	Template data.
prop_down	Proportion for downsampling. Default to 1 (no downsampling).

Value

List of Dataframes.

Examples

```
magicPred()
```

---

magicPred_all *magicPred_all*

Description

function to predict on plain test data (no hierarchy)

Usage

```
magicPred_all(list_test_data, magic_model, ref_model_info = NULL,
  n_cores = 1, ref_data_train = NULL, verbose = F)
```

Arguments

list_test_data	List of unlabeled dataframes. It has only the two columns of marker expression.
magic_model	Global trained model.
ref_model_info	Template model.
n_cores	Number of cores to use. Default to 1.
ref_data_train	Template data.
verbose	If True, show messages. Default to False.

Value

List of Dataframes.

Examples

```
magicPred_all()
```

magicPred_hierarchy *magicPred_hierarchy*

Description

function to predict the gates on the ungated .fcs samples.

Usage

```
magicPred_hierarchy(list_test_sets, list_models_local, df_tree,
  magic_model, list_local_train, n_cores = 1)
```

Arguments

list_test_sets	contains the list of root dataframe for each ungated fcs file imported.
list_models_local	contains the optimized local models pre-generated using the magicTrain_local function.
df_tree	contains the info related to the populations hierarchy.
magic_model	Global trained model.
list_local_train	contains the local training sets (gated data of the sample manually gated).
n_cores	Number of cores to use. Default to 1.

Value

List of Dataframes.

Examples

```
magicPred_hierarchy()
```

magicTrain *magicTrain*

Description

function to generate one training model based on a list of training sets (no hierarchy).

Usage

```
magicTrain(df_train, n_cores = 1, train_model = "rf", k_cv = 10,
  list_index = NULL, n_tree = 10, tune_lenght = 3,
  size_nnet_units = 100, decay_nnet = 0.1, method_control = "oob")
```

Arguments

df_train	training dataframe generated by the get_train_data function.
n_cores	Number of cores to use. Default to 1.
train_model	Type of training model Default to rf.
k_cv	Number of k for cross-validation (if method control=cv)
list_index	List of vector of indices to use in training generated by the get_indices_cross_val.
n_tree	Number of trees for random forest.
tune_lenght	Number of parameters values trained during cross-validation.
size_nnet_units	Number of units in hidden layer (if train_model=nnet).
decay_nnet	Decay parameter value for nnet model.
method_control	Type of training control: oob or cv. Default to oob.

Value

model object.

Examples

```
magicTrain()
```

---

magicTrain_dt *magicTrain_dt*

Description

function to generate a decision tree training model.

Usage

```
magicTrain_dt(Xtrain, Ytrain, k_cv = 10, list_index = NULL,
  tune_lenght = 10)
```

Arguments

Xtrain	Dataframe of training features.
Ytrain	Dataframe of labels (one column).
k_cv	Number of k for cross-validation (if method control=cv)
list_index	List of vector of indices to use in training.
tune_lenght	Number of parameters values trained during cross-validation.

Value

model object.

Examples

```
magicTrain_dt()
```

`magicTrain_local` *magicTrain_local*

Description

function to generate the local training models using the list of hierarchical training set.

Usage

```
magicTrain_local(list_train_sets, n_tree = 10, train_model = "rf",
  n_cores = 1)
```

Arguments

`list_train_sets`	List of labeled dataframe to train generated by the get_local_train function.
`n_tree`	Number of tree for random forest model.
`train_model`	Type of training model. Default to random forest ("rf").
`n_cores`	Number of cores to use. Default to 1.

Value

List of models objects.

Examples

```
magicTrain_local()
```

`magicTrain_nb` *magicTrain_nb*

Description

function to generate a naive bayes training model.

Usage

```
magicTrain_nb(Xtrain, Ytrain, k_cv = 10)
```

Arguments

Xtrain	Dataframe of training features.
Ytrain	Dataframe of labels (one column).
k_cv	Number of k for cross-validation (if method control=cv).

Value

model object.

Examples

```
magicTrain_nb()
```

---

magicTrain_nnet *magicTrain_nnet*

Description

function to generate a neural net training model.

Usage

```
magicTrain_nnet(Xtrain, Ytrain, k_cv = 10, list_index = NULL,
  size = 100, decay = 0.1)
```

Arguments

Xtrain	Dataframe of training features.
Ytrain	Dataframe of labels (one column).
k_cv	Number of k for cross-validation (if method control=cv).
list_index	List of vector of indices to use in training.
size	Number of units in hidden layer (if train_model=nnet).
decay	Decay parameter value for nnet model.

Value

model object.

Examples

```
magicTrain_nnet()
```

`magicTrain_rf` *magicTrain_rf*

Description

function to generate a random forest training model.

Usage

```
magicTrain_rf(Xtrain, Ytrain, list_index = NULL, n_tree = 10,
  method_control = "oob", k_cv = 10)
```

Arguments

`Xtrain`	Dataframe of training features.
`Ytrain`	Dataframe of labels (one column).
`list_index`	List of vector of indices to use in training.
`n_tree`	Number of trees for random forest.
`method_control`	Type of training control: oob or cv. Default to oob.
`k_cv`	Number of k for cross-validation (if method control=cv).

Value

model object.

Examples

```
magicTrain_rf()
```

`name_pop_gating` *name_pop_gating*

Description

function to get the name of all the pops of the gating hierarchy.

Usage

```
name_pop_gating(gh)
```

Arguments

`gh`	GatingHierarchy.

Value

Vector of characters.

Examples

```
name_pop_gating()
```

`post_process_gates` *post_process_gates*

Description

function to post process the events after model prediction.

Usage

```
post_process_gates(gated_df, n_cores = 1, thr_dist = 0.15,
  include_zero = F, remove_centroids = T, type = "dist",
  concavity_val = 5)
```

Arguments

`gated_df`	dataframe with labels (third column).
`n_cores`	Number of cores. Default to 1.
`thr_dist`	Distance threshold for centroids calculation. Default to 0.15.
`include_zero`	Consider centroid of label 0. Default to False.
`remove_centroids`	Remove centroids too near each other based on thr_dist value.
`type`	Type of post-processing.
`concavity_val`	Concavity of polygons for the "polygon" type of post-processing

Value

Dataframe.

Examples

```
post_process_gates()
```

pre_process_manual_binary *pre_process_manual_binary*

Description

In other words it generates the dataset that indicates what cells belong to the selected pop (1) 0 otherwise. The dataset is always the dataset of the Root pop. gh and pop are mandatory arguments, dim and mode have a default value.

Usage

```
pre_process_manual_binary(gh, pop)
```

Arguments

gh	GatingHierarchy.
pop	Name of the population to get events assignments.

Value

Dataframe.

Examples

```
pre_process_manual_binary()
```

process_test_data *process_test_data*

Description

function to get test data correctly formatted.

Usage

```
process_test_data(test_data, prop_down = 1)
```

Arguments

test_data	Dataframe of bivariate markers expression.
prop_down	Proportion of events (downsampling). Default to 1 (no downsampling).

Value

Dataframe.

Examples

```
process_test_data()
```

`range01` *range01*

Description

function to put data in range 0-1.

Usage

```
range01(x)
```

Arguments

`x`	Vector of numbers to scale.

Value

Vector of numbers.

Examples

```
range01()
```

`subsetting_binary_df` *subsetting_binary_df*

Description

function to get binary dataset with gate assignation based on the mother population of the selected pop (instead of the root).

Usage

```
subsetting_binary_df(gh, binary_df)
```

Arguments

`gh`	GatingHierarchy.
`binary_df`	Dataframe generated by the pre_process_manual_binary function.

Value

Dataframe.

Examples

```
subsetting_binary_df()
```

`update_label_association` *update_label_association*

Description

function to update label association.

Usage

```
update_label_association(df)
```

Arguments

`df`	Dataframe.

Value

Vector.

Examples

```
update_label_association()
```

Index

.densRange, 2

add_labels_column, 3
assign_events_to_nearest_centroids, 3

check_polygons_intersection, 4
compute_gates, 5
csv_to_dens, 5

exports_plots, 6
extract_polygon_gates, 7

get_centroids, 7
get_classes_expr_df, 8
get_density_features, 8
get_density_scores, 9
get_dist_template, 10
get_distance_loc_vs_test, 9
get_hierarchy_all_pops, 11
get_hull_all_gates, 11
get_indices_cross_val, 12
get_local_train_sets, 12
get_paths_training, 13
get_paths_training_v2, 14
get_pop_multiclass, 15
get_pops_hierarchy_list, 14
get_slot_hierarchy_list, 15
get_test_sets, 16
get_train_data, 16
get_weights_density_features, 17

import_png_image, 18
import_reference_csv, 18
import_sample_gated, 19
import_test_set, 19
import_test_set_csv, 20

magicPlot, 20
magicPred, 21
magicPred_all, 22
magicPred_hierarchy, 23
magicTrain, 23
magicTrain_dt, 24
magicTrain_local, 25
magicTrain_nb, 25
magicTrain_nnet, 26
magicTrain_rf, 27

name_pop_gating, 27

post_process_gates, 28
pre_process_manual_binary, 29
process_test_data, 29

range01, 30

subsetting_binary_df, 30

update_label_association, 31