

Package ‘scClassifR’

October 3, 2021

Type Package

Title Pretrained learning models for cell type prediction on single cell RNA-sequencing data

Version 1.0.0

Description The package comprises a set of pretrained machine learning models to predict basic immune cell types. This enables all users to quickly get a first annotation of the cell types present in their dataset without requiring prior knowledge. scClassifR also allows users to train their own models to predict new cell types based on specific research needs.

License MIT + file LICENSE

Encoding UTF-8

biocViews SingleCell, Transcriptomics, GeneExpression, SupportVectorMachine, Classification, Software

Imports dplyr, ggplot2, caret, ROCR, pROC, data.tree, methods, stats, e1071, ape, kernlab, utils

Suggests knitr, scRNAseq, testthat

VignetteBuilder knitr

Depends R (>= 4.1), Seurat, SingleCellExperiment, SummarizedExperiment

LazyData true

RoxygenNote 7.1.1

git_url <https://git.bioconductor.org/packages/scClassifR>

git_branch RELEASE_3_13

git_last_commit 2be48cc

git_last_commit_date 2021-05-19

Date/Publication 2021-10-03

Author Vy Nguyen [aut] (<<https://orcid.org/0000-0003-3436-3662>>),
Johannes Griss [cre] (<<https://orcid.org/0000-0003-2206-9511>>)

Maintainer Johannes Griss <johannes.griss@meduniwien.ac.at>

R topics documented:

cell_type	2
cell_type<-	3
checkObjectValidity	4
check_parent_child_coherence,Seurat-method	8
classify_cells	9
clf	11
default_models	12
delete_model	12
features	13
parent	14
plant_tree	15
plot_roc_curve	15
p_thres	16
p_thres<-	17
save_new_model	17
scClassifR	18
show,scClassifR-method	19
test_classifier	20
tirosh_mel80_example	22
train_classifier	23
Index	27

cell_type	<i>cell_type</i>
-----------	------------------

Description

Returns the cell type for the given classifier.

Usage

cell_type(classifier)

Arguments

classifier [scClassifR](#) object

Value

cell type of object

`cell_type<-`

3

Examples

```
data("tirosh_mel80_example")
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  cell_type = "B cells", features = selected_features_B)
cell_type(clf_b)
```

<code>cell_type<-</code>	<i>Setter for cell_type Change cell type for a classifier</i>
-----------------------------	---

Description

Setter for cell_type Change cell type for a classifier

Usage

```
cell_type(classifier) <- value
```

Arguments

<code>classifier</code>	scClassifR object. The object is returned from the train_classifier function.
<code>value</code>	the new cell type

Value

scClassifR object with the new cell type.

Examples

```
data("tirosh_mel80_example")
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")
cell_type(clf_b) <- "B cell"
```

checkObjectValidity *Internal functions of scClassifR package*

Description

Check if a scClassifR object is valid
Filter cells from ambiguous chars and non applicable cells in a [Seurat](#) object
Filter cells from ambiguous chars and non applicable cells in a [SingleCellExperiment](#) object
Construct a uniform tag vector for all forms of labels in a [Seurat](#) object
Construct a uniform tag vector for all forms of labels in a [SingleCellExperiment](#) object
Process parent classifier in a [Seurat](#) object
Process parent classifier in a [SingleCellExperiment](#) object

Usage

```
checkObjectValidity(object)

checkCellTypeValidity(cell_type)

checkFeaturesValidity(features)

checkParentValidity(parent)

checkPThresValidity(p_thres)

checkClassifierValidity(clf)

parent(classifier) <- value

clf(classifier) <- value

features(classifier) <- value

balance_dataset(mat, tag)

train_func(mat, tag)

transform_to_zscore(mat)

load_models(path_to_models)

select_features(mat, features)

check_parent_child_coherence(
  obj,
```

```
    pos_parent,
    parent_cell,
    cell_type,
    target_cell_type,
    ...
)

filter_cells(obj, tag_slot)

## S4 method for signature 'Seurat'
filter_cells(obj, tag_slot)

## S4 method for signature 'SingleCellExperiment'
filter_cells(obj, tag_slot)

construct_tag_vect(obj, cell_type, ...)

## S4 method for signature 'Seurat'
construct_tag_vect(obj, cell_type, tag_slot)

## S4 method for signature 'SingleCellExperiment'
construct_tag_vect(obj, cell_type, tag_slot)

process_parent_clf(
  obj,
  parent_tag_slot,
  parent_cell_type,
  parent_clf,
  path_to_models,
  zscore = TRUE,
  ...
)

## S4 method for signature 'Seurat'
process_parent_clf(
  obj,
  parent_tag_slot,
  parent_cell_type,
  parent_clf,
  path_to_models,
  zscore = TRUE,
  seurat_assay,
  seurat_slot,
  ...
)

## S4 method for signature 'SingleCellExperiment'
process_parent_clf(
```

```

    obj,
    parent_tag_slot,
    parent_cell_type,
    parent_clf,
    path_to_models,
    zscore = TRUE,
    sce_assay,
    ...
)

make_prediction(mat, classifier, pred_cells, ignore_ambiguous_result = TRUE)

simplify_prediction(meta.data, mat, classifiers)

verify_parent(mat, classifier, meta.data)

test_performance(mat, classifier, tag)

```

Arguments

object	The request classifier to check.
cell_type	name of cell type
features	list of selected features
parent	Classifier parent to check.
p_thres	Classifier probability threshold to check.
clf	Classifier to check.
classifier	classifier
value	the new classifier
mat	expression matrix
tag	tag of data
path_to_models	path to databases, or by default
obj	object
pos_parent	a vector indicating parent clf prediction
parent_cell	name of parent cell type
target_cell_type	alternative cell types (in case of testing clf)
...	arguments passed to other methods
tag_slot	tag slot in SingleCellExperiment object indicating cell type
parent_tag_slot	string, name of annotation tag slot in object indicating pre-assigned/predicted parent cell type
parent_cell_type	name of parent cell type

parent_clf	scClassifR object corresponding to classification model for the parent cell type
zscore	boolean indicating the transformation of gene expression in object to zscore or not
seurat_assay	name of assay to use in Seurat object
seurat_slot	type of expression data to use in Seurat object
sce_assay	name of assay to use in SingleCellExperiment object
pred_cells	a whole prediction for all cells
ignore_ambiguous_result	whether ignore ambiguous result
meta.data	object meta data
classifiers	classifiers

Value

TRUE if the classifier is valid or the reason why it is not
 TRUE if the cell type is valid or the reason why it is not.
 TRUE if the features is valid or the reason why it is not.
 TRUE if the parent is valid or the reason why it is not.
 TRUE if the p_thres is valid or the reason why it is not.
 TRUE if the classifier is valid or the reason why it is not.
 scClassifR object with the new parent.
 scClassifR object with the new trained classifier.
 scClassifR object with the new features.
 a list of balanced count matrix and corresponding tags of balanced count matrix
 the classification model (caret object)
 row wise center-scaled count matrix
 list of classifiers
 filtered matrix
 list of adjusted object and adjusted tag slot
 adjusted Seurat object
 adjusted SingleCellExperiment object
 a binary vector for cell tag
 list of cells which are positive to parent clf
 prediction
 simplified prediction
 applicable matrix
 clf performance

 check_parent_child_coherence, Seurat-method

Internal functions of scClassifR package

Description

Check label coherence in parent and child cell type in a [Seurat](#) object.

Check label coherence in parent and child cell type in a [SingleCellExperiment](#) object

Usage

```
## S4 method for signature 'Seurat'
check_parent_child_coherence(
  obj,
  pos_parent,
  parent_cell,
  cell_type,
  target_cell_type,
  tag_slot
)

## S4 method for signature 'SingleCellExperiment'
check_parent_child_coherence(
  obj,
  pos_parent,
  parent_cell,
  cell_type,
  target_cell_type,
  tag_slot
)
```

Arguments

obj	object
pos_parent	a vector indicating parent clf prediction
parent_cell	name of parent cell type
cell_type	name of cell type
target_cell_type	alternative cell types (in case of testing clf)
tag_slot	tag slot in SingleCellExperiment object indicating cell type

Value

TRUE if the classifier is valid or the reason why it is not

TRUE if the cell type is valid or the reason why it is not.

TRUE if the features is valid or the reason why it is not.
 TRUE if the parent is valid or the reason why it is not.
 TRUE if the p_thres is valid or the reason why it is not.
 TRUE if the classifier is valid or the reason why it is not.
 scClassifR object with the new parent.
 scClassifR object with the new trained classifier.
 scClassifR object with the new features.
 a list of balanced count matrix and corresponding tags of balanced count matrix
 the classification model (caret object)
 row wise center-scaled count matrix
 list of classifiers
 filtered matrix
 list of adjusted object and adjusted tag slot
 adjusted [Seurat](#) object
 adjusted [SingleCellExperiment](#) object
 a binary vector for cell tag
 list of cells which are positive to parent clf
 prediction
 simplified prediction
 applicable matrix
 clf performance

 classify_cells

Classify cells from multiple models

Description

Classify cells from multiple models

Usage

```

classify_cells(
  classify_obj,
  classifiers = NULL,
  cell_types = "all",
  path_to_models = c("default", "."),
  ignore_ambiguous_result = FALSE,
  ...
)

```

```

## S4 method for signature 'Seurat'
classify_cells(
  classify_obj,
  classifiers = NULL,
  cell_types = "all",
  path_to_models = c("default", "."),
  ignore_ambiguous_result = FALSE,
  seurat_assay = "RNA",
  seurat_slot = "counts",
  ...
)

## S4 method for signature 'SingleCellExperiment'
classify_cells(
  classify_obj,
  classifiers = NULL,
  cell_types = "all",
  path_to_models = c("default", "."),
  ignore_ambiguous_result = FALSE,
  sce_assay = "logcounts",
  ...
)

```

Arguments

<code>classify_obj</code>	the object containing cells to be classified
<code>classifiers</code>	list of classification models. The model is obtained from <code>train_classifier</code> function or available in current working space. Users may test the model using <code>test_classifier</code> before using this function. If classifiers contain classifiers for sub cell types, classifiers for parent cell type must be indicated first in order to be applied before children classifiers. If classifiers is <code>NULL</code> , the method will use all classifiers in database.
<code>cell_types</code>	list of cell types containing models to be used for classification, only applicable if the models have been saved to package.
<code>path_to_models</code>	path to the folder containing the list of models. As default value, the pretrained models in the package will be used. If user has trained new models, indicate the folder containing the <code>new_models.rda</code> file.
<code>ignore_ambiguous_result</code>	return all ambiguous predictions (multiple cell types) to empty. When this parameter turns to <code>TRUE</code> , most probably predicted cell types will be ignored.
<code>...</code>	arguments passed to other methods
<code>seurat_assay</code>	name of assay to use in Seurat object, defaults to 'RNA' assay.
<code>seurat_slot</code>	type of expression data to use in Seurat object. Some available types are: "counts", "data" and "scale.data". Default to "counts", which is unnormalized data.
<code>sce_assay</code>	name of assay to use in SingleCellExperiment object, defaults to 'logcounts' assay.

Value

the input object with new slots in cells meta data New slots are: predicted_cell_type, most_probable_cell_type, slots in form of [cell_type]_p and [cell_type]_class

Examples

```
# load small example dataset
data("tirosh_mel80_example")

# train one classifier for one cell type, for ex, B cell
# define genes to use to classify this cell type
selected_features_B = c("CD19", "MS4A1", "CD79A")

# train the classifier
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")

# do the same thing with other cell types, for example, T cells
selected_features_T = c("CD4", "CD8A", "CD8B")
set.seed(123)
clf_t <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_T, cell_type = "T cells")

# create a list of classifiers
classifier_ls <- list(clf_b, clf_t)

# classify cells with list of classifiers
seurat.obj <- classify_cells(classify_obj = tirosh_mel80_example,
  classifiers = classifier_ls)
```

clf

clf

Description

Returns the classifier of the [scClassifR](#) object

Usage

```
clf(classifier)
```

Arguments

classifier [scClassifR](#) object

Value

Classifier is the object returned by caret SVM learning process. More information about the caret package: <https://topepo.github.io/caret/>

Examples

```
data("tirosh_mel80_example")
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")
clf(clf_b)
```

default_models	<i>Pretrained classifiers for human cells</i>
----------------	---

Description

Pretrained classifier obtained by training Sade-Feldman dataset and testing on Jerby-Arnon dataset.

Usage

```
default_models
```

Format

a list of `scClassifR` objects

Author(s)

Vy Nguyen, November 2020

delete_model	<i>Delete model/branch from package</i>
--------------	---

Description

Delete model/branch from package

Usage

```
delete_model(cell_type, path.to.models = ".")
```

Arguments

`cell_type` string indicating the cell type of which the model will be removed from package
 Attention: deletion of a parent model will also delete all of child model.

`path.to.models` path to the folder containing the list of models in which the to-be-deleted model is.

Value

no return value, but the model is deleted from database

Examples

```
# load small example dataset
data("tirosh_mel80_example")

# train a classifier
set.seed(123)
selected_features_T = c("CD4", "CD8A", "CD8B")
clf_t <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_T, cell_type = "t cells")

# save a classifier to system
save_new_model(new_model = clf_t)

# delete classifier from system
delete_model("t cells")
```

features	<i>features</i>
----------	-----------------

Description

Returns the set of features for the given classifier.

Usage

```
features(classifier)
```

Arguments

`classifier` scClassifR object

Value

Applied features of object

Examples

```
data("tirosh_mel80_example")
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")
features(clf_b)
```

parent

parent

Description

Returns the parent of the cell type corresponding to the given classifier.

Usage

```
parent(classifier)
```

Arguments

classifier scClassifR object

Value

Parent model of object

Examples

```
data("tirosh_mel80_example")
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")
parent(clf_b)
```

plant_tree	<i>Plant tree from list of models</i>
------------	---------------------------------------

Description

Plant tree from list of models

Usage

```
plant_tree(models.file.path = c("default", "."))
```

Arguments

models.file.path
list of models. If not provided, list of default pretrained models in the package will be used.

Value

tree structure and plot of tree

Examples

```
# to create the tree of classifiers  
# (in this example, based on default classifiers)  
plant_tree()
```

plot_roc_curve	<i>Plot roc curve</i>
----------------	-----------------------

Description

Plot roc curve

Usage

```
plot_roc_curve(test_result)
```

Arguments

test_result result of test_classifier function

Value

ggplot2 roc curve

Examples

```
# load small example dataset
data("tirosh_mel80_example")

# train a classifier, for ex: B cell
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")

clf_b_test <- test_classifier(test_obj = tirosh_mel80_example,
  classifier = clf_b)

# run plot curve on the test result
roc_curve <- plot_roc_curve(test_result = clf_b_test)
```

*p_thres**p_thres*

Description

Returns the probability threshold for the given classifier.

Usage

```
p_thres(classifier)
```

Arguments

classifier scClassifR object

Value

Predicting probability threshold of object

Examples

```
data("tirosh_mel80_example")
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")
p_thres(clf_b)
```

p_thres<- *Setter for predicting probability threshold*

Description

Setter for predicting probability threshold

Usage

```
p_thres(classifier) <- value
```

Arguments

classifier	scClassifR object. The object is returned from the train_classifier function.
value	the new threshold

Value

scClassifR object with the new threshold.

Examples

```
data("tirosh_mel80_example")
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")
clf_b_test <- test_classifier(test_obj = tirosh_mel80_example,
  classifier = clf_b)
# assign a new threshold probability for prediction
p_thres(clf_b) <- 0.4
```

save_new_model *Save a model to the package*

Description

Save a model to the package

Usage

```
save_new_model(new_model, include.default = TRUE, path.to.models = ".")
```

Arguments

`new_model` new model to be added into the classification tree

`include.default` whether include the default models of the package in the list of new trained models or not. If users further want to classify cells, they can only use default pretrained model list or their new model list. Including default models in new trained models helps users using both of them once. In addition, default pretrained models of the package cannot be changed or removed. This can be done with the new trained model list.

`path.to.models` path to the folder containing the list of new models.

Value

no return value, but the model is now saved to database

Examples

```
# load small example dataset
data("tirosh_mel80_example")

# train classifier
selected_features_T = c("CD4", "CD8A", "CD8B")
set.seed(123)
clf_t <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_T, cell_type = "t cells")

# save the trained classifier to system
# (test classifier can be used before this step)
save_new_model(new_model = clf_t, path.to.models = ".")

# verify if new model has been saved
print(names(load("../new_models.rda")))
delete_model("t cells")
```

scClassifR

scClassifR class.

Description

This class is returned by the `train_classifier` function. Generally, scClassifR objects are never created directly.

Usage

```
scClassifR(cell_type, clf, features, p_thres, parent)

scClassifR(cell_type, clf, features, p_thres, parent)
```

Arguments

cell_type	character. Name of the cell type.
clf	list. Trained model returned by caret train function.
features	vector/character containing features used for the training.
p_thres	numeric. Probability threshold for the cell type to be assigned for a cell.
parent	character. Parent cell type.

Value

A scClassifR object.

Slots

cell_type	character. Name of the cell type.
clf	list. Trained model returned by caret train function.
features	vector/character containing features used for the training.
p_thres	numeric. Probability threshold for the cell type to be assigned for a cell.
parent	character. Parent cell type.

Examples

```
# load small example dataset
data("tirosh_mel80_example")

# train a classifier, for ex: B cell
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
                          features = selected_features_B, cell_type = "B cells")

clf_b
```

```
show,scClassifR-method
```

Show object

Description

Show object

Usage

```
## S4 method for signature 'scClassifR'
show(object)
```

Arguments

object scClassifR object

Value

print to console information about the object

Examples

```
data("tirosh_mel80_example")
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")
clf_b
```

test_classifier	<i>Testing process.</i>
-----------------	-------------------------

Description

Testing process.

Usage

```
test_classifier(
  test_obj,
  classifier,
  target_cell_type = NULL,
  parent_clf = NULL,
  path_to_models = c("default", "."),
  zscore = TRUE,
  ...
)

## S4 method for signature 'Seurat,scClassifR'
test_classifier(
  test_obj,
  classifier,
  target_cell_type = NULL,
  parent_clf = NULL,
  path_to_models = c("default", "."),
  zscore = TRUE,
  seurat_tag_slot = "active.ident",
  seurat_parent_tag_slot = "predicted_cell_type",
  seurat_assay = "RNA",
```

```

    seurat_slot = "counts",
    ...
)

## S4 method for signature 'SingleCellExperiment,scClassifR'
test_classifier(
  test_obj,
  classifier,
  target_cell_type = NULL,
  parent_clf = NULL,
  path_to_models = c("default", "."),
  zscore = TRUE,
  sce_tag_slot = "ident",
  sce_parent_tag_slot = "predicted_cell_type",
  sce_assay = "logcounts",
  ...
)

```

Arguments

test_obj	xxobject that can be used for testing
classifier	classification model
target_cell_type	vector indicating other cell types than cell labels that can be considered as the main cell type in classifier, for example, c("plasma cell", "b cell", "b cells", "activating b cell"). Default as NULL.
parent_clf	scClassifR object corresponding to classification model for the parent cell type
path_to_models	path to the folder containing the list of models. As default, the pretrained models in the package will be used. If user has trained new models, indicate the folder containing the new_models.rda file.
zscore	boolean, whether gene expression is transformed to zscore
...	arguments passed to other methods
seurat_tag_slot	string, name of annotation slot indicating cell tag/label in the testing object. Strings indicating cell types are expected in this slot. For Seurat object, default value is "active.ident". Expected values is string or binary/logical, 0/"no"/F/FALSE: not being new cell type, 1/"yes"/T/TRUE: being new cell type.
seurat_parent_tag_slot	string, name of tag slot in cell meta data indicating pre-assigned/predicted parent cell type. Default field is "predicted_cell_type". The slot must contain only string values.
seurat_assay	name of assay to use in Seurat object, defaults to 'RNA' assay.
seurat_slot	type of expression data to use in Seurat object. Some available types are: "counts", "data" and "scale.data". Default to "counts", which contains unnormalized data.

sce_tag_slot	string, name of annotation slot indicating cell tag/label in the testing object. Strings indicating cell types are expected in this slot. Default value is "ident". Expected values are string or binary/logical, 0/"no"/F/FALSE: not being new cell type, 1/"yes"/T/TRUE: being new cell type.
sce_parent_tag_slot	string, name of tag slot in cell meta data indicating pre-assigned/predicted parent cell type. Default is "predicted_cell_type". The slot must contain only string values.
sce_assay	name of assay to use in SingleCellExperiment object, defaults to 'logcounts' assay.

Value

result of testing process in form of a list, including predicted values, prediction accuracy at a probability threshold, and roc curve information.

Note

Only one cell type is expected for each cell. Ambiguous cell type, such as: "T cells/NK cells/ILC", will be ignored. Subtypes used in testing model for parent cell types can be indicated as parent cell type, or can be indicated in target_cell_type. For example, when testing for B cells, plasma cells can be annotated as B cells, or target_cell_type is set c("plasma cells").

Examples

```
# load small example dataset
data("tirosh_mel80_example")

# train the classifier
selected_features_B = c("CD19", "MS4A1", "CD79A")
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "B cells")

# test the classifier, target cell type can be in other formats or
# alternative cell type that can be considered as the classified cell type
clf_b_test <- test_classifier(test_obj = tirosh_mel80_example,
  classifier = clf_b, target_cell_type = c("B cell"))
clf_b_test
```

tirosh_mel80_example *A Seurat Object Sample*

Description

A Seurat object sample was made from the dataset GSE72056, samples corresponding to patient CY80.

Usage

tirosh_mel80_example

Format

a [Seurat](#) object

Author(s)

Itay Tirosh, 2016-04-05

Source

WEIZMANN INSTITUTE OF SCIENCE

train_classifier	<i>Train cell type classifier</i>
------------------	-----------------------------------

Description

Train a classifier for a new cell type. If cell type has a parent, only available for [scClassifR](#) object as parent cell classifying model.

Usage

```
train_classifier(  
  train_obj,  
  cell_type,  
  features,  
  parent_cell = NA_character_,  
  parent_clf = NULL,  
  path_to_models = c("default", "."),  
  zscore = TRUE,  
  balance = TRUE,  
  ...  
)  
  
## S4 method for signature 'Seurat'  
train_classifier(  
  train_obj,  
  cell_type,  
  features,  
  parent_cell = NA_character_,  
  parent_clf = NULL,  
  path_to_models = c("default", "."),  
  zscore = TRUE,  
  balance = TRUE,
```

```

    seurat_tag_slot = "active.ident",
    seurat_parent_tag_slot = "predicted_cell_type",
    seurat_assay = "RNA",
    seurat_slot = "counts",
    ...
)

## S4 method for signature 'SingleCellExperiment'
train_classifier(
  train_obj,
  cell_type,
  features,
  parent_cell = NA_character_,
  parent_clf = NULL,
  path_to_models = c("default", "."),
  zscore = TRUE,
  balance = TRUE,
  sce_tag_slot = "ident",
  sce_parent_tag_slot = "predicted_cell_type",
  sce_assay = "logcounts",
  ...
)

```

Arguments

train_obj	object that can be used for training the new model. Seurat object or SingleCellExperiment object is expected. If the training model has parent, parent_tag_slot may have been indicated. This field would have been filled out automatically if user prece- dently run classify_cells function. If no (predicted) cell type annotation pro- vided, the function can be run if 1- parent_cell or 2- parent_clf is provided.
cell_type	string indicating the name of the subtype This must exactly match cell tag/label if cell tag/label is a string.
features	list of features used for the new training model
parent_cell	string indicated the name of the parent cell type, if parent cell type classifier has already been saved in model database. Adjust path_to_models for exact database.
parent_clf	classification model for the parent cell type
path_to_models	path to the folder containing the model database. As default, the pretrained models in the package will be used. If user has trained new models, indicate the folder containing the new_models.rda file.
zscore	whether gene expression in train_obj is transformed to zscore
balance	whether applying balancing on training set before training
...	arguments passed to other methods
seurat_tag_slot	string, name of slot in cell meta data indicating cell tag/label in the training object. Strings indicating cell types are expected in this slot. For Seurat ob-

	ject, default value is "active.ident". Expected values are string or binary/logical, 0/"no"/F/FALSE: not being new cell type, 1/"yes"/T/TRUE: being new cell type.
seurat_parent_tag_slot	string, name of a slot in cell meta data indicating assigned/predicted cell type. Default is "predicted_cell_type". This slot would have been filled automatically if user have called classify_cells function. The slot must contain only string values.
seurat_assay	name of assay to use in training object. Default to 'RNA' assay.
seurat_slot	type of expression data to use in training object. For Seurat object, available types are: "counts", "data" and "scale.data". Default to "counts", which contains unnormalized data.
sce_tag_slot	string, name of annotation slot indicating cell tag/label in the training object. For SingleCellExperiment object, default value is "ident". Expected values are string or binary/logical, 0/"no"/F/FALSE: not being new cell type, 1/"yes"/T/TRUE: being new cell type.
sce_parent_tag_slot	string, name of a slot in cell meta data indicating pre-assigned/predicted cell type. Default field is "predicted_cell_type". This field would have been filled automatically when user called classify_cells function. The slot must contain only string values.
sce_assay	name of assay to use in training object. Default to 'logcounts' assay.

Value

[scClassifR](#) object

Note

Only one cell type is expected for each cell in object. Ambiguous cell type, such as: "T cells/NK cells/ILC", will be ignored from training. Subtypes used in training model for parent cell types must be indicated as parent cell type. For example, when training for B cells, plasma cells must be annotated as B cells in order to be used.

Examples

```
# load small example dataset
data("tirosh_mel80_example")

# this dataset already contains pre-defined cell labels
table(Seurat::Idents(tirosh_mel80_example))

# define genes to use to classify this cell type (B cells in this example)
selected_features_B = c("CD19", "MS4A1", "CD79A")

# train the classifier, the "cell_type" argument must match
# the cell labels in the data, except upper/lower case
set.seed(123)
clf_b <- train_classifier(train_obj = tirosh_mel80_example,
  features = selected_features_B, cell_type = "b cells")
```

```
# classify cell types using B cell classifier,
# a test classifier process may be used before applying the classifier
tirosh_mel80_example <- classify_cells(classify_obj = tirosh_mel80_example,
classifiers = c(clf_b))

# tag all cells that are plasma cells (random example here)
tirosh_mel80_example[['plasma_cell_tag']] <- c(rep(1, 80), rep(0, 400))

# set new features for the subtype
p_features = c("SDC1", "CD19", "CD79A")

# train the classifier, the "B cell" classifier is used as parent.
# This means, only cells already classified as "B cells" will be evaluated.
# the "tag_slot" parameter tells the classifier to use this cell meta data
# for the training process.
set.seed(123)
plasma_clf <- train_classifier(train_obj = tirosh_mel80_example,
cell_type = "Plasma cell", features = p_features, parent_clf = clf_b,
seurat_tag_slot = 'plasma_cell_tag')
```

Index

* **datasets**
 default_models, 12
 tirosh_mel80_example, 22
balance_dataset (checkObjectValidity), 4
cell_type, 2
cell_type<-, 3
check_parent_child_coherence
 (checkObjectValidity), 4
check_parent_child_coherence, Seurat-method, 8
check_parent_child_coherence, SingleCellExperiment-method
 (check_parent_child_coherence, Seurat-method), 8
checkCellTypeValidity
 (checkObjectValidity), 4
checkClassifierValidity
 (checkObjectValidity), 4
checkFeaturesValidity
 (checkObjectValidity), 4
checkObjectValidity, 4
checkParentValidity
 (checkObjectValidity), 4
checkPThresValidity
 (checkObjectValidity), 4
classify_cells, 9
classify_cells, Seurat-method
 (classify_cells), 9
classify_cells, SingleCellExperiment-method
 (classify_cells), 9
clf, 11
clf<-(checkObjectValidity), 4
construct_tag_vect
 (checkObjectValidity), 4
construct_tag_vect, Seurat-method
 (checkObjectValidity), 4
construct_tag_vect, SingleCellExperiment-method
 (checkObjectValidity), 4
default_models, 12
delete_model, 12
features, 13
features<-(checkObjectValidity), 4
filter_cells (checkObjectValidity), 4
filter_cells, Seurat-method
 (checkObjectValidity), 4
filter_cells, SingleCellExperiment-method
 (checkObjectValidity), 4
load_models (checkObjectValidity), 4
make_prediction (checkObjectValidity), 4
p_thres, 16
p_thres<-, 17
parent, 14
parent<-(checkObjectValidity), 4
plant_tree, 15
plot_roc_curve, 15
process_parent_clf
 (checkObjectValidity), 4
process_parent_clf, Seurat-method
 (checkObjectValidity), 4
process_parent_clf, SingleCellExperiment-method
 (checkObjectValidity), 4
save_new_model, 17
scClassifR, 2, 7, 11, 12, 18, 21, 23, 25
select_features (checkObjectValidity), 4
Seurat, 4, 7–10, 21, 23–25
show, scClassifR-method, 19
simplify_prediction
 (checkObjectValidity), 4
SingleCellExperiment, 4, 6–10, 22, 24, 25
test_classifier, 20
test_classifier, Seurat, scClassifR-method
 (test_classifier), 20
test_classifier, SingleCellExperiment, scClassifR-method
 (test_classifier), 20

`test_performance (checkObjectValidity),`
 4
`tirosh_mel80_example,` 22
`train_classifier,` 18, 23
`train_classifier, Seurat-method`
 (`train_classifier`), 23
`train_classifier, SingleCellExperiment-method`
 (`train_classifier`), 23
`train_func (checkObjectValidity),` 4
`transform_to_zscore`
 (`checkObjectValidity`), 4

`verify_parent (checkObjectValidity),` 4