Package 'ensembleTax'

July 22, 2025

Title Ensemble Taxonomic Assignments of Amplicon Sequencing Data

Version 1.1.1

Author Dylan Catlett [aut, cre], Kevin Son [ctb], Connie Liang [ctb]

Maintainer Dylan Catlett <dcat4444@gmail.com>

Description Creates ensemble taxonomic assignments of amplicon sequencing data in R using outputs of multiple taxonomic assignment algorithms and/or reference databases. Includes flexible algorithms for mapping taxonomic nomenclatures onto one another and for computing ensemble taxonomic assignments.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports dplyr, Biostrings, DECIPHER, stringr, ggplot2, reshape2, usethis

Depends R (>= 2.10)

Suggests knitr, rmarkdown, markdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2021-05-21 16:00:03 UTC

Contents

assign.ensembleTax 2
bayes.sample 5
bayestax2df 5
gg_13_8_train_set_97 7
idtax.pr2.sample 7
idtax.silva.sample 8
idtax2df 8

pr2v4.12.0 10
rdp_train_set_16 11
rubric.sample 11
silva.nr.v138 12
sort_my_taxtab 12
synonyms_v2 13
taxmapper 14

Index **17**

`assign.ensembleTax` *Computes ensemble taxonomic assignments for each ASV in an amplicon data set*

Description

Computes ensemble taxonomic assignments for each ASV in an amplicon data set

Usage

```
assign.ensembleTax(
  x,
  tablenames = names(x),
  ranknames = c("kingdom", "supergroup", "division", "class", "order", "family",
    "genus", "species"),
  weights = rep(1, length(x)),
  tiebreakz = NULL,
  count.na = TRUE,
  assign.threshold = 0
)
```

Arguments

`x`	A list of dataframes of type character or list (no factors) that contain an arbitrary number of meta-data columns (e.g. ASV sequences or numbers), and other columns named according to ranknames that include taxonomic assignments for each ASV in the data set
`tablenames`	A character vector of the names of each taxonomy table provided in x. Default is names(x)
`ranknames`	The names of ranks (columns) of the taxonomy tables included in x. These are used to track ASV-identifying data through the ensemble calculations.
`weights`	A numeric vector with length = length(x) that specifies relative weights to the taxonomic assignments in the corresponding element of x. Default is a vector with all elements =1 to specify equal weighting of all taxonomy tables assignments. All values must be integers.

`tiebreakz` NULL is the default. Alternatively, a character vector containing the tablenames in order of priority to be used as a tie-breaker in the event that multiple taxonomic names are found at equal (weighted) highest frequencies (above assign.threshold).

`count.na` TRUE or FALSE indicating whether you would like NA assignments considered in the ensemble calculation. TRUE considers NA assignments, FALSE does not consider NA assignments. assign.threshold is implemented differently depending on whether this is TRUE or FALSE.

`assign.threshold` A number between 0 and 1 that indicates the (weighted) proportion at which a particular taxonomic name must be assigned in the input taxonomy tables in order to be assigned to the ensemble taxonomic assignment. When count.na=FALSE, proportions are calculated only relative to the number of tables with no NA assignments. When count.na=TRUE, proportions are calculated relative to the sum of the weights argument.

Details

The algorithm takes as input a list of taxonomy tables (dataframes of type character or list; no factors) and assumes rows correspond to ASVs/OTUs and columns correspond to taxonomic assignments at ranks listed in descending order in the input ranknames. All taxonomy tables should follow the same taxonomic nomenclature (naming and ranking conventions), should include ASV/OTU-identifying columns (e.g. ASV sequences or a column of asv numbers, etc), and each row of each taxonomy table should represent the same ASV/OTU. Use of the functions bayestax2df, idtax2df, and/or taxmapper will ensure your taxonomy tables meet these requirements. Be advised that rownames of each taxonomy table are set to NULL by assign.ensembleTax.

Ensemble taxonomic assignments are computed by finding the highest-frequency taxonomic assignment for each ASV across all input taxonomy tables. Several parameters can be controlled by the user to weight the assignments of specific taxonomy tables more highly than others (weights), to favor assignments by a specific table in the event that multiple assignments are found at the same (weighted) highest frequency (tiebreakz), to set a (weighted) frequency threshold above which a taxonomic assignment must be found to be assigned in the ensemble (assign.threshold), and finally to ignore non-assignments signalled by NA in the frequency and assignment computations (count.na).

The output is a dataframe of ASVs and corresponding ensemble taxonomic assignments.

Value

a dataframe containing ensemble taxonomic assignments

Author(s)

Dylan Catlett

Kevin Son

See Also

idtax2df, bayestax2df, taxmapper

Examples

```
fake1.pr2 <- data.frame(ASV = c("AAAA", "ATCG", "GCGC", "TATA", "TCGA"),
         kingdom = c("Eukaryota", "Eukaryota", "Eukaryota", "Eukaryota",
         "Eukaryota"),
         supergroup = c(NA, "Stramenopiles", "Rhizaria", "Stramenopiles",
         "Alveolata"),
         division = c(NA, "Ochrophyta", "Radiolaria", "Opalozoa",
         "Dinoflagellata"),
         class = c(NA, "Bacillariophyta", "Polycystinea", "MAST-12",
         "Syndiniales"),
         order = c(NA, "Bacillariophyta_X", "Collodaria", "MAST-12A", NA),
         family = c(NA, "Polar-centric-Mediophyceae", "Collophidiidae", NA,
         NA),
         genus = c(NA, NA, "Collophidium", NA, NA),
         species = as.character(c(NA, NA, NA, NA, NA)),
         stringsAsFactors = FALSE)
fake2.pr2 <- data.frame(ASV = c("AAAA", "ATCG", "GCGC", "TATA", "TCGA"),
         kingdom = c("Eukaryota", "Eukaryota", "Eukaryota", "Eukaryota",
         "Eukaryota"),
         supergroup = c(NA, "Stramenopiles", "Rhizaria", "Stramenopiles",
         "Alveolata"),
         division = c(NA, "Opalozoa", "Radiolaria", "Opalozoa",
         "Dinoflagellata"),
         class = c(NA, NA, "Polycystinea", NA, "Dinophycese"),
         order = c(NA, NA, "Collodaria", NA, NA),
         family = c(NA, NA, "Collophidiidae", NA, NA),
         genus = c(NA, NA, "Collophidium", NA, NA),
         species = as.character(c(NA, NA, NA, NA, NA)),
         stringsAsFactors = FALSE)
head(fake1.pr2)
head(fake2.pr2)
xx <- list(fake1.pr2, fake2.pr2)
names(xx) <- c("fake1", "fake2")
xx
eTax <- assign.ensembleTax(xx,
           tablenames = names(xx),
           ranknames = c("kingdom", "supergroup", "division","class","order",
           "family","genus","species"),
           tiebreakz = NULL,
           count.na=TRUE,
           assign.threshold = 0.5,
           weights=rep(1,length(xx)))
head(eTax)
eTax <- assign.ensembleTax(xx,
                   tablenames = names(xx),
                   ranknames = c("kingdom", "supergroup", "division",
                   "class","order","family","genus","species"),
                   tiebreakz = NULL,
                   count.na=FALSE,
                   assign.threshold = 0.5,
                   weights=c(2,1))
head(eTax)
```

`bayes.sample` *Example output of dada2 assignTaxonomy function*

Description

Example output of dada2 assignTaxonomy function

Usage

```
bayes.sample
```

Format

^ list with 2 elements

tax taxonomic assignments

boot bootstrap confidence estimates

`bayestax2df` *Converts the output of DADA2's assignTaxonomy, which implements a naive bayesian classifier, into a dataframe compatible with the algorithms used in ensembleTax*

Description

Converts the output of DADA2's assignTaxonomy, which implements a naive bayesian classifier, into a dataframe compatible with the algorithms used in ensembleTax

Usage

```
bayestax2df(
  tt,
  db = "pr2",
  ranks = NULL,
  boot = 0,
  rubric = NULL,
  return.conf = FALSE
)
```

Arguments

tt	The taxonomy table output by DADA2's assignTaxonomy function.
db	The database you ran assignTaxonomy against. Either "pr2", "silva", "rdp", or "gg" are supported. You may set to NULL and include a character vector of rank (column) names for other databases.
ranks	NULL, or a character vector of column names if db is set to NULL
boot	The bootstrap threshold below which taxonomic assignments should be set to NA. This can also be done with DADA2's assignTaxonomy but is included here for convenience.
rubric	NULL, or a DNAStringSet (see Biostrings package) with ASV sequences named by your preferred ASV identifier. Both the ASV sequence and identifier will be merged with the output dataframe. If NULL, ASV-identifying data are excluded in the output dataframe.
return.conf	If TRUE, returns a list where the first element is your formatted taxonomy table and the second element is a dataframe of bootstrap confidence values. If FALSE, your formatted taxonomy table is returned as a dataframe.

Details

For consistency with dada2's assignTaxonomy function, when used with Silva, RDP, or GreenGenes it subsamples the ranks c("domain", "phylum", "class", "order", "family", "genus"). Set db = NULL and supply ranks for databases that aren't directly supported. If a rubric is supplied with ASV-identifying meta-data (this is highly recommended), the output taxonomy table is sorted by the (first returned column of) ASV-identifying data.

Value

a dataframe formatted for use with taxmapper and/or ensembleTax

Author(s)

Dylan Catlett

Connie Liang

See Also

idtax2df, ensembleTax, taxmapper

Examples

```
data("bayes.sample")
data("rubric.sample")
head(bayes.sample)
head(rubric.sample)
df <- bayestax2df(tt = bayes.sample, db = "pr2", boot = 0, rubric = NULL,
return.conf = FALSE)
head(df)
df <- bayestax2df(tt = bayes.sample, db = "pr2", boot = 0,
```

```
rubric = rubric.sample, return.conf = FALSE)
head(df)
df <- bayestax2df(tt = bayes.sample, db = "pr2", boot = 60,
rubric = rubric.sample, return.conf = FALSE)
head(df)
df <- bayestax2df(tt = bayes.sample, db = "pr2", boot = 60,
rubric = rubric.sample, return.conf = TRUE)
head(df)
```

`gg_13_8_train_set_97` *All unique taxonomic assignments from the GreenGenes v13.8 clusted at 97%*

Description

All unique taxonomic assignments from the GreenGenes v13.8 clusted at 97%

Usage

```
gg_13_8_train_set_97
```

Format

^ dataframe with 4163 rows and 7 columns

domain domain assignments

phylum phylum assignments

class class assignments

order order assignments

family family assignments

genus genus assignments

species genus assignments

`idtax.pr2.sample` *Example output of DECIPHER idtaxa function with pr2 taxonomy*

Description

Example output of DECIPHER idtaxa function with pr2 taxonomy

Usage

```
idtax.pr2.sample
```

Format

^ list with 5 elements

1 tax data for ASV 1

2 tax data for ASV 2

3 tax data for ASV 3

4 tax data for ASV 4

5 tax data for ASV 5

`idtax.silva.sample` *Example output of DECIPHER idtaxa function with silva taxonomy*

Description

Example output of DECIPHER idtaxa function with silva taxonomy

Usage

```
idtax.silva.sample
```

Format

^ list with 5 elements

1 tax data for ASV 1

2 tax data for ASV 2

3 tax data for ASV 3

4 tax data for ASV 4

5 tax data for ASV 5

`idtax2df` *Converts outputs of DECIPHER's idtaxa algorithm into a dataframe compatible with the algorithms used in ensembleTax.*

Description

Converts outputs of DECIPHER's idtaxa algorithm into a dataframe compatible with the algorithms used in ensembleTax.

Usage

```
idtax2df(
  tt,
  db = "pr2",
  ranks = NULL,
  boot = 0,
  rubric = NULL,
  return.conf = FALSE
)
```

Arguments

tt	The taxonomy table output by DECIPHER's idtaxa algorithm
db	The database you ran idtaxa against. Either "pr2", "silva", "rdp", or "gg" are supported.
ranks	NULL, or a character vector of column names if db is set to NULL
boot	The bootstrap threshold below which taxonomic assignments should be set to NA. This can also be done with DECIPHER's idtaxa but is included here for convenience.
rubric	a DNAStringSet (see Biostrings package) with ASV sequences named by your preferred ASV identifier. Both the ASV sequence and identifier will be merged with the output dataframe. If NULL, ASV-identifying data is not included in the output dataframe.
return.conf	If TRUE, returns a list where the first element is your formatted taxonomy table and the second element is a dataframe of bootstrap confidence values. If FALSE, your formatted taxonomy table is returned as a dataframe.

Details

For consistency with DADA2's assignTaxonomy function, when used with Silva, RDP, or GreenGenes it subsamples the ranks c("domain", "phylum", "class", "order", "family", "genus"). Set db = NULL and supply ranks for databases that aren't directly supported. The output taxonomy table is sorted by the ASV-identifying data supplied in the rubric.

CAUTION: the idtaxa algorithm does not return any ASV-identifying data in its output "taxon" object. The elements of tt should thus be supplied in the same order as the elements in rubric. This will typically be the case so long as there is no tampering with the rubric or taxon object in between implementing idtaxa and their use here.

Value

a dataframe formatted for use with taxmapper and/or ensembleTax

Author(s)

Dylan Catlett

Connie Liang

See Also

bayestax2df, ensembleTax, taxmapper

Examples

```
data("idtax.pr2.sample")
data("rubric.sample")
head(idtax.pr2.sample)
head(rubric.sample)
df <- idtax2df(tt = idtax.pr2.sample, db = "pr2", ranks = NULL, boot = 0,
rubric = NULL, return.conf = FALSE)
head(df)
df <- idtax2df(tt = idtax.pr2.sample, db = "pr2", ranks = NULL, boot = 0,
rubric = rubric.sample, return.conf = FALSE)
head(df)
df <- idtax2df(tt = idtax.pr2.sample, db = "pr2", ranks = NULL, boot = 60,
rubric = rubric.sample, return.conf = FALSE)
head(df)
df <- idtax2df(tt = idtax.pr2.sample, db = "pr2", ranks = NULL, boot = 60,
rubric = rubric.sample, return.conf = TRUE)
head(df)
```

---

pr2v4.12.0 *All unique taxonomic assignments from the pr2 reference database v4.12.0*

Description

All unique taxonomic assignments from the pr2 reference database v4.12.0

Usage

```
pr2v4.12.0
```

Format

^ dataframe with 45352 rows and 8 columns

kingdom kingdom assignments

supergroup supergroup assignments

division division assignments

class class assignments

order order assignments

family family assignments

genus genus assignments

species species assignments

rdp_train_set_16 *All unique taxonomic assignments from the RDP Train Set 16*

Description

All unique taxonomic assignments from the RDP Train Set 16

Usage

```
rdp_train_set_16
```

Format

^ dataframe with 2472 rows and 6 columns

domain domain assignments

phylum phylum assignments

class class assignments

order order assignments

family family assignments

genus genus assignments

rubric.sample *Example rubric with ASV-identifying data*

Description

Example rubric with ASV-identifying data

Usage

```
rubric.sample
```

Format

^ DNAStringSet with 5 elements

sv1 sample ASV 1

sv2 sample ASV 2

sv3 sample ASV 3

sv4 sample ASV 4

sv5 sample ASV 5

silva.nr.v138 *All unique taxonomic assignments from the Silva SSU nr database v138*

Description

All unique taxonomic assignments from the Silva SSU nr database v138

Usage

```
silva.nr.v138
```

Format

^ dataframe with 6011 rows and 6 columns

domain domain assignments

phylum phylum assignments

class class assignments

order order assignments

family family assignments

genus genus assignments

sort_my_taxtab *Sorts taxonomy table by ASV-identifying columns.*

Description

Sorts taxonomy table by ASV-identifying columns.

Usage

```
sort_my_taxtab(tt, ranknames)
```

Arguments

tt	A taxonomy table supplied as a dataframe (no factors)
ranknames	A character vector of the names of columns of tt that contain taxonomic assignments. tt is sorted by columns not included in ranknames.

Details

A helper function for the ...2df family of pre-processing functions. If multiple columns are available to sort, it uses the left-most column.

Value

a dataframe sorted by the columns specified in ranknames

Author(s)

Dylan Catlett

See Also

bayestax2df, idtax2df

Examples

```
data("bayes.sample")
data("rubric.sample")
bayes.pretty <- bayestax2df(bayes.sample, rubric = rubric.sample)
sort_my_taxtab(bayes.pretty,
ranknames = c("kingdom", "supergroup", "division", "class", "order",
"family", "genus", "species"))
```

---

synonyms_v2 *Taxonomic synonyms searched by the taxmapper algorithm*

Description

Taxonomic synonyms searched by the taxmapper algorithm

Usage

```
synonyms_v2
```

Format

^ dataframe with 174 rows and 11 columns

Name_1 first synonym

Name_2 second synonym

Name_3 third synonym

Name_4 fourth synonym

Name_5 fifth synonym

Name_6 sixth synonym

Name_7 seventh synonym

References Reference for some synonyms

Notes.References Notes from references

X Additional references for some synonyms

X.1 Additional references for some synonyms

`taxmapper` *Maps an input taxonomy table onto a different taxonomic nomenclature.*

Description

Maps an input taxonomy table onto a different taxonomic nomenclature.

Usage

```
taxmapper(
  tt,
  tt.ranks = colnames(tt),
  tax2map2 = "pr2",
  exceptions = c("Archaea", "Bacteria"),
  ignore.format = FALSE,
  synonym.file = "default",
  streamline = TRUE,
  outfilez = NULL
)
```

Arguments

`tt`	The input taxonomy table you would like to map onto a new taxonomic nomenclature. Should be a dataframe of type char or list (no factors).
`tt.ranks`	A character vector of the column names where taxonomic names are found in tt. Supply them heirarchically (e.g. kingdom –> species)
`tax2map2`	The taxonomic nomenclature you would like to map onto. pr2 v4.12.0, Silva SSU v138 nr, GreenGenes v13.8 clustered at 97% similarity, and the RDP train set v16 are included in the ensembleTax package. You can map to these by specifying "pr2", "Silva", "gg", or "rdp". Otherwise should be a dataframe of type character or list (no factors) with each column corresponding to a taxonomic rank.
`exceptions`	A character vector of taxonomic names at the basal/root rank of tt that will be propagated onto the mapped taxonomy. ASVs assigned to these names will retain these names at their basal/root rank in the mapped taxonomy. All other ranks are assigned NA.
`ignore.format`	If TRUE, the algorithm modifies taxonomic names in tt to account for common variations in taxonomic name syntax and/or formatting commonly encountered in reference databases (e.g. Pseudo-nitzschia will map to Pseudonitzschia). If FALSE, formatting issues may preclude mapping of synonymous taxonomic names (e.g. Pseudo-nitzschia will NOT map to Pseudonitzschia). An exhaustive list of formatting details is included in Details. Note that formatting variants are only generated for the names in tt. This can cause some issues for mapping in the other direction (e.g. Pseudonitzschia in tt will NOT map to Pseudo-nitzschia in tax2map2 whether or not ignore.format is TRUE).

`synonym.file` If "default", taxmapper uses taxonomic synonyms included with the ensembleTax package. If a custom taxonomic synonym file is preferred, a string corresponding to the name of the csv file should be supplied. Taxonomic synonyms are searched when exact name matches are not found in tax2map2. ignore.format applies to synonyms if TRUE. Specify NULL if you wish to forego synonym searches.

`streamline` If TRUE, only the mapped version of tt is returned as a dataframe. If FALSE, a 3-element list is returned where element 1 is the mapping key returned as a dataframe, element 2 is a character vector of all names that could not be mapped (no exact matches found in tax2map2), and element 3 is the mapped version of tt (a dataframe).

`outfilez` If NULL, mapping files are not saved to the current working directory. Otherwise should be a 3-element character vector including, in this order, the name of the file to store the taxonomic mapping key, the name of the file to store the names that could not be mapped, and the name of the file to store the ASVs supplied with tt with their mapped taxonomic assignments. Each element of the vector should end in csv (only csv files may be saved)

Details

Exceptions should be used when the user knows a particular taxonomic group is not found in tax2map2. The user is responsible for supplying valid taxonomic names as these must be found in tt and will be propagated as given to all ASVs that are assigned this name in tt. This should only be used for high-level taxonomic groups that are not found in a database (e.g. for retaining Eukaryota when mapping onto a prokaryote-only taxonomic nomenclature).

When ignore.format = TRUE, names for which taxmapper cannot find exact matches in tax2map2 are altered in case an exact match was not found due to formatting issues. To do this taxmapper first removes square brackets ("[]"). It then checks for hyphens "-", underscores "_", and single spaces " ". If these are found, variants of the name with the hyphen/underscore/spaces replaced by each of the other two, as well as all subnames spearated by these characters, and all subnames pasted together with none of these special characters, are searched against tax2map2 for exact matches. It also creates all-lower and all-upper case versions of these elements and again searches for exact name matches in tax2map2. Words generated by this process that are 2 characters or less are not searched for matches in tax2map2. All alternative names created when ignore.format = TRUE are also searched for synonyms in synonym.file if specified.

To prevent matching of arbitrary names often used in reference databases (eg, "Clade_X"), and after creating all of the above alternative names if ignore.format = TRUE, those that BEGIN with any of the words below are are not use in exact name matching. Instead, the lowest assigned non-ambiguous name is determined (any name that begins with a word NOT included in the list below) and is appended to the ambiguous name separated by a hyphen. The words taxmapper flags as ambiguous are: "Clade", "CLADE", "clade", "Group", "GROUP", "group", "Class", "CLASS", "class", "Subclass", "SubClass", "SUBCLASS", "subclass", "Subclade", "SubClade", "SUBCLADE", "subclade", "Subgroup", "SubGroup", "SUBGROUP", "subgroup", "Sub group", "Sub Group", "SUB GROUP", "sub group", "Sub clade", "Sub Clade", "SUB CLADE", "sub clade", "Sub class", "Sub Class", "SUB CLASS", "sub class", "Sub_group", "Sub_Group", "SUB_GROUP", "sub_group", "Sub_clade", "Sub_Clade", "SUB_CLADE", "sub_clade", "Sub_class", "Sub_Class", "SUB_CLASS", "sub_class", "Sub-group", "Sub-Group", "SUB-GROUP", "sub-group", "Sub-clade", "Sub-Clade", "SUB-CLADE", "sub-clade", "Sub-class", "Sub-Class", "SUB-CLASS", "sub-class", "incertae sedis",

"INCERTAE SEDIS", "Incertae sedis", "Incertae Sedis", "incertae-sedis", "INCERTAE-SEDIS", "Incertae-sedis", "Incertae-Sedis", "incertae_sedis", "INCERTAE_-SEDIS", "Incertae_sedis", "Incertae_Sedis", "incertaesedis", "INCERTAESEDIS", "Incertaesedis", "IncertaeSedis", "unclassified", "UNCLASSIFIED", "Unclassified", "Novel", "novel", "NOVEL", "sp", "sp.", "spp", "spp.", "lineage", "Lineage", "LINEAGE"

For high-throughput implementation of taxmapper, it's recommended to set streamline = TRUE.

Value

If streamline = TRUE, a dataframe formatted for use with ensembleTax that contains mapped taxonomic assignments for each ASV/OTU in the data set.

If streamline = FALSE, a 3-element list where the first element is a dataframe that contains all unique input taxonomic assignments and their corresponding mapped outputs, the second element is a character vector that contains all taxonomic names that could not be mapped, and the third element contains mapped taxonomic assignments for each ASV in the data set.

If is.null(outfilez) = FALSE, three csv files are saved in the current working directory containing each of the three list elements above.

Author(s)

Dylan Catlett

Kevin Son

See Also

idtax2df, bayestax2df, ensembleTax

Examples

```
fake.silva <- data.frame(ASV = c("AAAA", "ATCG", "GCGC", "TATA", "TCGA"),
domain = c("Bacteria", "Eukaryota", "Eukaryota", "Eukaryota", "Eukaryota"),
phylum = c("Firmicutes", "Diatomea", "Retaria", "MAST-12", "Diatomea"),
class = c(NA, "Coscinodiscophytina_cl", "Polycystinea", "MAST-12A",
"Mediophyceae"),
order = c(NA, "Fragilariales", "Collodaria", NA, NA),
family = c(NA, "Fragilariales_fa", "Collodaria_fa", NA, NA),
genus = c(NA, "Podocystis", "Collophidium", NA, NA),
stringsAsFactors = FALSE)
head(fake.silva)
mapped.silva <- taxmapper(fake.silva,
                          tt.ranks = colnames(fake.silva)[2:ncol(fake.silva)],
                          tax2map2 = "pr2",
                          exceptions = c("Archaea", "Bacteria"),
                          ignore.format = FALSE,
                          synonym.file = "default",
                          streamline = TRUE,
                          outfilez = NULL)
```

Index

∗ **datasets**
bayes.sample, 5
gg_13_8_train_set_97, 7
idtax.pr2.sample, 7
idtax.silva.sample, 8
pr2v4.12.0, 10
rdp_train_set_16, 11
rubric.sample, 11
silva.nr.v138, 12
synonyms_v2, 13

assign.ensembleTax, 2

bayes.sample, 5
bayestax2df, 5

gg_13_8_train_set_97, 7

idtax.pr2.sample, 7
idtax.silva.sample, 8
idtax2df, 8

pr2v4.12.0, 10

rdp_train_set_16, 11
rubric.sample, 11

silva.nr.v138, 12
sort_my_taxtab, 12
synonyms_v2, 13

taxmapper, 14