Package ‘supersigs’

May 2, 2024

Title  Supervised mutational signatures
Version  1.12.0
Date  2021-12-02
Depends  R (>= 4.1)
Imports  assertthat, caret, dplyr, tidyr, rsample, methods, rlang, utils, Biostrings, stats, SummarizedExperiment
Suggests  BSgenome.Hsapiens.UCSC.hg19, BSgenome.Hsapiens.UCSC.hg38, knitr, rmarkdown, ggplot2, testthat, VariantAnnotation
Description  Generate SuperSigs (supervised mutational signatures) from single nucleotide variants in the cancer genome. Functions included in the package allow the user to learn supervised mutational signatures from their data and apply them to new data. The methodology is based on the one described in Afsari (2021, ELife).
biocViews  FeatureExtraction, Classification, Regression, Sequencing, WholeGenome, SomaticMutation
BugReports  https://github.com/TomasettiLab/supersigs/issues
URL  https://tomasettilab.github.io/supersigs/
License  GPL-3
Encoding  UTF-8
LazyData  true
LazyDataCompression  gzip
RoxygenNote  7.1.1
VignetteBuilder  knitr
Config/testthat/edition  3
git_url  https://git.bioconductor.org/packages/supersigs
git_branch  RELEASE_3_19
git_last_commit  132e64c
git_last_commit_date  2024-04-30
Repository  Bioconductor 3.19
Date/Publication  2024-05-01
example_dt

Example dataset of mutations

Description
A dataset containing a list of mutations and other necessary attributes

Usage

example_dt

Format
A data frame with 10 rows and 5 columns:

sample_id  ID of the patient
age  age of the patient
chromosome  chromosomal position of the mutation
position  position of the mutation
ref  original nucleotide
alt  mutated nucleotide
### get_signature

**Function to obtain a SuperSig**

**Description**

Generate a tissue-specific SuperSig for a given dataset of mutations and exposure factor. Returns the SuperSig and a classification model trained with the SuperSig.

**Usage**

```r
get_signature(data, factor, wgs = FALSE)
```

**Arguments**

- **data**
  - a data frame of mutations containing columns for `sample_id`, `age`, `IndVar`, and the 96 trinucleotide mutations (see vignette for details).

- **factor**
  - the factor/exposure (e.g. "age", "smoking"). If the factor = "age", the SuperSig is computed using counts. Otherwise, rates (counts/age) are used.

- **wgs**
  - logical value indicating whether sequencing data is whole-genome (`wgs = TRUE`) or whole-exome (`wgs = FALSE`).

**Value**

`get_signature` returns an object of class `SuperSig`.

**Examples**

```r
head(example_dt) # use example data from package
input_dt <- make_matrix(example_dt) # convert to correct format
input_dt$IndVar <- c(1, 1, 1, 0, 0) # add IndVar column
get_signature(data = input_dt, factor = "Age") # get SuperSig
```

### make_matrix

**Function to transform mutations into "matrix" format**

**Description**

Transform a data frame of mutations in long format into a data frame of trinucleotide mutations with flanking bases in a wide matrix format.

**Usage**

```r
make_matrix(data, genome = "hg19")
```
Arguments

- `data`: a data frame of mutations in VCF format (see vignette for details)
- `genome`: the reference genome used ("hg19" or "hg38")

Value

`make_matrix` returns a data frame of mutations, one row per sample

Examples

```r
head(example_dt) # use example data from package
going <- make_matrix(example_dt) # convert to correct format
head(input_dt)
```

---

`partial_signature` Function to remove the contribution of a SuperSig

Description

Remove the contribution of a SuperSig from the data and return the data.

Usage

`partial_signature(data, object)`

Arguments

- `data`: a data frame of mutations containing columns for `sample_id`, `age`, `IndVar`, and the 96 trinucleotide mutations (see vignette for details)
- `object`: an object of class SuperSig

Value

`predict_signature` returns the original data frame with the contribution of a supervised signature removed

Examples

```r
head(example_dt) # use example data from package
going <- make_matrix(example_dt) # convert to correct format
going$IndVar <- c(1, 1, 1, 0, 0) # add IndVar column
supersig <- get_signature(data = going, factor = "Age") # get SuperSig
partial_signature(data = going, object = supersig)
```
### predict_signature

**Function to predict using SuperSig object**

#### Description

Using a generated SuperSig, predict on a new dataset and return predicted probabilities for each observation.

#### Usage

```r
predict_signature(object, newdata, factor)
```

#### Arguments

- `object`: an object of class `SuperSig`
- `newdata`: a data frame of mutations containing columns for `sample_id`, `age`, `IndVar`, and the 96 trinucleotide mutations (see vignette for details)
- `factor`: the factor/exposure (e.g. "age", "smoking")

#### Value

`predict_signature` returns the original data frame with additional columns for the feature counts and classification score.

#### Examples

```r
head(example_dt) # use example data from package
input_dt <- make_matrix(example_dt) # convert to correct format
input_dt$IndVar <- c(1, 1, 1, 0, 0) # add IndVar column
out <- get_signature(data = input_dt, factor = "Age") # get SuperSig
newdata <- predict_signature(out, newdata = input_dt, factor = "age")
suppressPackageStartupMessages({library(dplyr)})
head(newdata %>% select(score))
```

### process_vcf

**Function to transform VCF object into "matrix" format**

#### Description

Transform a VCF object into a data frame of trinucleotide mutations with flanking bases in a wide matrix format. The function assumes that the VCF object contains only one sample and that each row in rowRanges represents an observed mutation in the sample.

#### Usage

```r
process_vcf(vcf)
```
Arguments

vcf a VCF object (from VariantAnnotation package)

Value

process_vcf returns a data frame of mutations, one row per mutation

Examples

# Use example vcf from VariantAnnotation
suppressPackageStartupMessages({library(VariantAnnotation)})
fl <- system.file("extdata", "chr22.vcf.gz", package="VariantAnnotation")
vcf <- VariantAnnotation::readVcf(fl, "hg19")

# Subset to first sample
vcf <- vcf[, 1]
# Subset to row positions with homozygous or heterozygous alt positions <- geno(vcf)$GT != "0|0"
vcf <- vcf[positions[, 1],]
colData(vcf)$age <- 50 # Add patient age to colData (optional)

# Run function
dt <- process_vcf(vcf)
head(dt)

simplify_signature Function to simplify signature representation into interpretable labels for visualization purposes

Description

Take a signature representation from SuperSig and group trinucleotides within each feature into interpretable labels, with optional IUPAC labeling from IUPAC_CODE_MAP in the Biostrings package

Usage

simplify_signature(object, iupac)

Arguments

object an object of class SuperSig
iupac logical value indicating whether to use IUPAC labels (iupac = TRUE) or not (iupac = FALSE)
**SuperSig-class**

**Value**

`simplify_signature` returns a vector of simplified features and their difference in mean mean rates between exposed and unexposed (or average rate if the factor is "age")

**Examples**

```r
head(example_dt) # use example data from package
input_dt <- make_matrix(example_dt) # convert to correct format
input_dt$IndVar <- c(1, 1, 1, 0, 0) # add IndVar column
supersig <- get_signature(data = input_dt, factor = "Smoking")
simplify_signature(object = supersig, iupac = FALSE)
simplify_signature(object = supersig, iupac = TRUE)
```

---

SuperSig-class  
An S4 class for SuperSig

---

**Description**

An S4 class for SuperSig

**Slots**

- **Signature** data frame of features and their difference in mean rates between exposed and unexposed (or the average rate if the factor is "age")
- **Features** list of features that comprise the signature and their representation in terms of the fundamental (trinucleotide) mutations
- **AUC** length-one numeric vector of the apparent AUC (i.e. not cross-validated)
- **Model** list of a glm class for trained logistic regression model

---

supersig_ls  
Trained SuperSigs from TCGA

---

**Description**

A list containing 67 SuperSigs

**Usage**

```r
supersig_ls
```

**Format**

A named list with 67 elements, each of which is a `SuperSig`
Index

* datasets
  example_dt, 2
  supersig_ls, 7

example_dt, 2
get_signature, 3
make_matrix, 3
partial_signature, 4
predict_signature, 5
process_vcf, 5

simplify_signature, 6
SuperSig (SuperSig-class), 7
SuperSig-class, 7
supersig_ls, 7