Package 'SigMA'

October 31, 2018

Title Signature Multivariate Analysis

Version 1.0.0.0

Description SigMA is a signature analysis tool optimized to detect the mutational signature associated to HR defect, Signature 3, from hybrid capture panels, exomes and whole genome sequencing. For panels with low SNV counts, conventional signature analysis tools do not perform well while the novel approach of SigMA allows it to detect Signature 3-positive tumors with 74% sensitivity at 10% false positive rate. One novelty of SigMA is a likelihood based matching: We associate a new patient's mutational spectrum to subtypes of tumors according to their signature composition. The subtypes of tumors are defined using the WGS data from ICGC and TCGA consortia, by a clustering of signature fractions with hierarchical clustering. The likelihood of the sample to belong to each tumor subtype is calculated, and the likelihood of Signature 3 is the sum of the likelihoods of all Signature 3-positive tumor subtypes. The second novel step is the multivariate analysis with gradient boosting machines, which allows us to obtain a final score for presence of Signature-3 combining likelihood with cosine similarity and exposure of Signature 3 obtained with non-negativel-least-squares (NNLS) algorithm. The multivariate analysis allows us to automatically handle different sequencing platforms. For different platforms different methods for signature analysis become more efficient, e.g. for WGS data it is not necessary to associate the tumor to a subtype of tumors, because it is possible to determine Signature 3 with NNLS acurately. We have a new feature also for these cases and we calculate the likelihood of NNLS decomposition to be unique. This likelihood value was found to be the most influential feature in the multivariate analysis.

Depends R (>= 3.4.0)

License What license is it under?

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

Imports BSgenome,
BSgenome.Hsapiens.UCSC.hg19,
GenomicRanges,
ggplot2,
gbm,
grid,
gridExtra,
IRanges,
nnls,
reshape2,
Rmisc,
VariantAnnotation

R topics documented:

`assignment` *Assigns a boolean based on a threshold on the likelihood or mva score for whether the signature is identified*

Description

Assigns a boolean based on a threshold on the likelihood or mva score for whether the signature is identified

Usage

```
assignment(df_in, method = "mva", signame = "Signature_3",
  data = NULL, tumor_type = "breast", do_strict = T, weight_cf)
```

Arguments

`df_in`	input data.frame
`method`	'median_catalog' for likelihood based selection or 'mva' for multivariate analysis score based selection
`signame`	name of the signature that user wants to identify, 'Signature_3' or 'Signature_msi'
`data`	'msk', 'seqcap' or 'wgs'
`tumor_type`	tumor type as listed in https://github.com/parklab/SigMA/ because the thresholds are tumor_type specific
`do_strict`	sets whether a strict threshold should be applied or a loose one

Value

a data.frame with a single column which contains the boolean indicating the presence of the signature

`calc_llh` *Calculates likelihood of the genome with respect to the available signature probability distributions*

Description

Calculates likelihood of the genome with respect to the available signature probability distributions

Usage

```
calc_llh(spectrum, signatures, counts = NULL, normalize = T)
```

Arguments

`spectrum`	is the mutational spectrum
`signatures`	is the reference signature catalog with the probability distributions
`counts`	is the number of cases in each cluster that is represented in the catalog. They are used as weights for each signature in the catalog
`normalize`	is true by default only for when it is used together with NNLS in the match_to_catalog function it is not normalized here but outside of the function

`cosine` *calculates cosine similarity between the spectrum and a set of signatures*

Description

calculates cosine similarity between the spectrum and a set of signatures

Usage

```
cosine(x, y)
```

Arguments

`spectrum`	is the mutational spectrum
`signatures`	is the reference signature catalog with the probability distribution

`decompose` *Decomposes the mutational spectrum of a genome in terms of tumor type specific signatures that were calculated through analysis of public WGS samples from ICGC and TCGA consortia, and contained as a list in the package. Non-negative-least squares algorithm is used and the number of signatures to be considered in the decomposition is increased gradually, first all pairs from among the available signatures are considered and minimal error pair is kept. Then all 3-signature combinations, 4-signature combinations and so on are considered. The result is updated if the error is smaller with larger number of signatures*

Description

Decomposes the mutational spectrum of a genome in terms of tumor type specific signatures that were calculated through analysis of public WGS samples from ICGC and TCGA consortia, and contained as a list in the package. Non-negative-least squares algorithm is used and the number of signatures to be considered in the decomposition is increased gradually, first all pairs from among the available signatures are considered and minimal error pair is kept. Then all 3-signature combinations, 4-signature combinations and so on are considered. The result is updated if the error is smaller with larger number of signatures

Usage

```
decompose(spect, signatures, data)
```

Arguments

`spect`	composite spectrum that is being decomposed
`signatures`	a data.frame that contains the signatures in its columns
`data`	sequencing platform that as in run(), used for setting the maximum number of signatures that is allowed in the decomposition

`lite_df` *produces a data.frame with fewer columns for easier use it is used in run.R function when lite_format is set to T*

Description

produces a data.frame with fewer columns for easier use it is used in run.R function when lite_format is set to T

Usage

```
lite_df(merged_output)
```

Arguments

`merged_output`	is the input data.frame

make_matrix *Converts somatic mutation call files in a directory either in the form of vcf or maf into a 96-dimensional matrix, it works for general number of context and for 1 or 2 strands*

Description

Converts somatic mutation call files in a directory either in the form of vcf or maf into a 96-dimensional matrix, it works for general number of context and for 1 or 2 strands

Usage

```
make_matrix(directory, file_type = "vcf",
  ref_genome = BSgenome.Hsapiens.UCSC.hg19::BSgenome.Hsapiens.UCSC.hg19,
  ncontext = 3, nstrand = 1, chrom_colname = NULL,
  pos_colname = NULL, ref_colname = NULL, alt_colname = NULL)
```

Arguments

directory	pointer to the directory where input vcf maf files reside
file_type	'maf', 'vcf' or 'custom'
ref_genome	name of the BSgenome currently set by default to BSgenome.Hsapiens.UCSC.hg19
ncontext	number of bases in the nucleotide sequence which makes up the spectrum, default 3
nstrand	number of strands to be considered, 1 contracts to a single strand which for ncontext = 3 gives the commonly used 96 dimensions
chrom_colname	used only for custom files a character string defining the colname which holds the chromosome number
pos_colname	used only for custom files a character string defining the colname which holds the position information
ref_colname	used only for custom files a character string defining the colname which holds the ref allele
alt_colname	used only for custom files a character string defining the colname which holds the alt allele

Examples

```
by default runs on vcf input and produces 96 dimensional spectra
make_matrix(directory = 'input')
make_matrix(directory = 'input',
            file_type = 'vcf',
            ref_genome = BSgenome.Hsapiens.UCSC.hg19,
            ncontext = 5,
            nstrand = 2)
```

match_to_catalog *Calculates the compatibility of a list of genomes to an input catalog based on likelihood and cosine similarity*

Description

Calculates the compatibility of a list of genomes to an input catalog based on likelihood and cosine similarity

Usage

```
match_to_catalog(genomes, signatures, data, cluster_fractions = NULL,
  method = "median_catalog")
```

Arguments

genomes	a data table or matrix with snv spectra in the first ntype columns and genomes in each row
signatures	the input catalog, a data table with signature spectra in each column
data	sets the type of sequencing platform used, options are 'msk', 'seqcap', 'wgs'
method	can be 'median_catalog', 'weighted_catalog' 'cosine_simil' or 'decompose. 'median_atalog' uses the signature catalog formed by clustering genome SNV spectra and using it as a probability distribution. The 'median_catalog' method can be used with any custom signatures data frame if the user intends to provide their own signature table.

Value

A data frame that contains the input genomes and in addition columns associated to each signature in in the catalog with likelihood and cosine simil values

plot_detailed *Generates a detailed plot per sample*

Description

Generates a detailed plot per sample

Usage

```
plot_detailed(file = NULL, sample = NULL)
```

Arguments

file	the csv file produced by SigMA

plot_summary *Generates summary plot*

Description

Generates summary plot

Usage

```
plot_summary(file = NULL)
```

Arguments

file	the csv file produced by SigMA

plot_tribase_dist *plots the 96 dimensional mutational spectrum*

Description

plots the 96 dimensional mutational spectrum

Usage

```
plot_tribase_dist(df_snvs, file_name = "test.png", labely = "N SNVs",
  legend = T, text_size = 10, signame = "")
```

Arguments

df_snvs	a data frame with 96-dimensional spectra on its columns
file_name	the name of the plot to be generated with the proper extension e.g. "test.pdf", "test.png", etc
labely	string for the label of the y axis
legend	boolean determining whether legend should be printed
text_size	size of the text of the x and y axis text and titles
signame	a text to be printed on the figure

`predict_mva` *This function uses the trained MVA, in particular gradient boosting models, inside the package to assign a probability for the existence of the signature of interest.*

Description

This function uses the trained MVA, in particular gradient boosting models, inside the package to assign a probability for the existence of the signature of interest.

Usage

```
predict_mva(input, signame, data, tumor_type = "breast", weight_cf)
```

Arguments

`input`	is a data frame that has likelihood cosine similarity and total snv values in it's columns
`signame`	name of the signature which is being identified
`data`	determines the sequencing platform see run()
`tumor_type`	tumor type tag see ?run

Value

a data.frame with a single column with the score of MVA

`run` *Runs SigMA: (1) calculates likelihood, cosine similarity, NNLS exposures, and likelihood of the decomposition. (2) These features are later used in multivariate analysis. (3) Based on scores a final decision on existence of the signature.*

Description

Runs SigMA: (1) calculates likelihood, cosine similarity, NNLS exposures, and likelihood of the decomposition. (2) These features are later used in multivariate analysis. (3) Based on scores a final decision on existence of the signature.

Usage

```
run(genome_file, output_file = NULL, do_assign = T, data = "msk",
  tumor_type = "breast", do_mva = T, check_msi = F, weight_cf = T,
  lite_format = F)
```

Arguments

genome_file	a csv file with snv spectra info can be created from vcf file using @make_genome_matrix() function see ?make_genome_matrix
output_file	the output file name, can be NULL in which case input file name is used and appended with "_output"
do_assign	boolean for whether a cutoff should be applied to determine the final decision or just the features should be returned
data	the options are "msk" (for a panel that is similar size to MSK-Impact panel with 410 genes), "seqcap" (for whole exome sequencing), or "wgs" (for whole genome sequencing)
tumor_type	the options are "bladder", "bone_other" (Ewing's sarcoma or Chordoma), "breast", "crc", "eso", "gbm", "lung", "lymph", "medullo", "osteo", "ovary", "panc_ad", "panc_en", "prost", "stomach", "thy", or "uterus". The exact correspondance of these names can be found in https://github.com/parklab/SigMA
do_mva	a boolean for whether multivariate analysis should be run
check_msi	is a boolean which determines whether the user wants to identify micro-sattelite instable tumors
weight_cf	is a boolean that determines whether number of tumors in each cluster is going to be used as weights in calculating probability, it only works for panels, for other platforms it is always T
lite_format	is a boolean when set T the output file is saved in the lite format with fewer columns for easier use by default it is F

Examples

```
run(genome_file = "input_genomes.csv",
    data = "msk",
    tumor_type = "ovary")
run(genome_file = "input_genomes.csv",
    data = "seqcap",
    tumor_type = "bone_other")
```

Index