Package 'MSIseq'

June 15, 2015

Maintainer Mini Huang <mini.huang@nus.edu.sg>

License GPL (>= 2)

Title Assess Tumor Microsatellite Instability with a Decision Tree Classifier from Exome Somatic Mutations

Type Package

Author Mini Huang

Description A decision tree classifier for detecting microsatellite instability (MSI) in somatic mutation data from whole exome sequencing. MSI is detected based on different mutation rates in all sites as well as in simple sequence repeats. This mechanism can also be applied to sequence data of targeted gene panels with shorter sequence length.

SystemRequirements Java

Version 1.0.0

biocViews Classification, SomaticMutation, Sequencing

Depends IRanges, RWeka, rJava, R.utils, R (>= 3.2.0)

Date 2015-06-10

NeedsCompilation no

Repository CRAN

Date/Publication 2015-06-15 16:26:10

R topics documented:

`MSIseq-package` *Assess Tumor Microsatellite Instability with a Decision Tree Classifier from Tumor Exome Somatic Mutations*

Description

A decision tree classifier for detecting microsatellite instability (MSI) in somatic mutation data from whole exome sequencing. MSI is detected based on different mutation rates in all sites as well as in simple sequence repeats. This mechanism can also be applied to sequence data of targeted gene panels with shorter sequence length.

Details

Package:	MSIseq
Type:	Package
Depends:	RWeka, rJava, R.utils, R (>= 3.2.0)
Imports:	RWeka, rJava, R.utils
Version:	1.0.0
Date:	2014-06-10

This package contains two main functions:

`MSIseq.train()` is used to generate a detector for MSI status from mutation information. `MSIseq.classify()` uses a detector to classify the MSI status of new tumors from mutation catalogs.

Author(s)

Mini Huang

Maintainer: Mini Huang <mini.huang@nus.edu.sg>

See Also

`MSIseq.train`, `MSIseq.classify`

`Compute.input.variables` *Compute Mutation Counts from Tumor Exome Somatic Mutations*

Description

This function returns a data frame of mutation counts from the mutation information in the `data` argument.

Usage

```
Compute.input.variables(data, repeats, uniform.seq.len=38, seq.len = NULL)
```

Arguments

`data`	A data frame in the NGS "mutation annotation file" format. See `NGStestdata` for details.
`repeats`	A data frame indicating the genome coordinates of simple sequence repeats.
`uniform.seq.len`	The length of the capture sequence. This argument should be used when the capture sequences for all the samples have the same length. This argument will be ignored when `seq` is supplied.
`seq.len`	A data frame with two columns: Tumor_Sample_Barcode and the corresponding Sequence_Length (Mb). This should be provided if the capture sequences for the tumors have different lengths.

Details

This function computes 9 variables (listed below) from `data`. Mutations are one of two types: single nucleotide substitutions (SNSs) and short insertions/deletions (indels).

- T.sns: total count of SNSs/Mb
- S.sns: count of SNSs in simple sequence repeats/Mb
- T.ind: total count of indels/Mb
- S.ind: count of indels in simple sequence repeats/Mb
- T: total mutation count/Mb
- S: mutation count in simple sequence repeats/Mb
- ratio.sns: S.sns/T.sns
- ratio.ind: S.ind/T.ind
- ratio: S/T

Value

A data frame with 9 columns that are the 9 variables listed above.

Author(s)

Mini Huang

See Also

MSIseq.train, MSIseq.classify,

Examples

```
## Not run:
## load sample data: NGStestdata, NGStestseqLen

data(NGStestdata)
data(NGStestseqLen)

## download the Hg19repeats annotation file and load it
url <-
"http://steverozen.net/data/Hg19repeats.rda"
file <- basename(url)
download.file(url, file)
load("Hg19repeats.rda")

## get mutation counts for test data

test.mutationNum<-Compute.input.variables(NGStestdata,
repeats=Hg19repeats, seq.len = NGStestseqLen)

## End(Not run)
```

---

find.mono.repeats *Find Mono-nucleotide Repeats*

Description

This function reads in a file with multiple fasta records and find the mononucleotide repeats in the text.

Usage

```
find.mono.repeats(text, min.len = 5)
```

Arguments

text	The return object from function read.fasta.
min.len	The minimum length of the mononucleotide repeats searched by the function.

Details

This function find the mononucleotide repeats in the text and return a data frame with the following columns.

- chr: a character vector that indicates the chromosome identifier with the "chr" prefix: "chr1", "chr2", ..., "chr22", "chrX", "chrY"
- start: a numeric vector that indicates the start position in the chromosome
- stop: a numeric vector that indicates the end position in the chromosome

Value

A data frame with 3 columns that are the 3 variables listed above.

Author(s)

Mini Huang

See Also

`read.fasta`

`MSIseq.classify` *Classify Microsatellite Instability with MSIseq Classifier*

Description

This function gives a classification result for MSI status based on `classifier` with the mutation information in the `mutationNum` argument.

Usage

```
MSIseq.classify(mutationNum, classifier = NGSclassifier, cancerType = NULL)
```

Arguments

`mutationNum`	A data frame output from `Compute.input.variables`, which containing 9 variables: T.sns, S.sns, T.ind, S.ind, T, S, Ratio.sns, Ratio.ind, Ratio.
`classifier`	A RWeka J48 model returned from the function `MSIseq.train`. Notice that if this argument is missing, the function will use a default build-in classifier, NGSclassifier.
`cancerType`	A data frame with two columns: Tumor_Sample_Barcode (tumor ID) and the corresponding cancer_type. Check `NGStraintype` for detail. If the classifier used in this function is trained with cancerType argument, cancerType should be provided here.

Details

This function gives a classification of MSI status for the samples with their `mutationNum` information. The classification is made with the decision tree model in `classifier`. It also flags the samples with likely POLE deficiency based on the criteria of high T.sns (> 60/Mb) and low S.ind (< 0.18/Mb).

Value

A data frame with three columns: Tumor_Sample_Barcode, the corresponding classified MSI_status, and a third column indicating whether the sample is likely POLE deficient.

Author(s)

Mini Huang

References

Kurt Hornik, Christian Buchta, Achim Zeileis (2009) Open-Source Machine Learning: R Meets Weka. Computational Statistics, 24(2), 225-232.

See Also

`MSIseq.train`, `Compute.input.variables`

Examples

```
## load sample data: test.mutationNum

data(test.mutationNum)
data(NGStestseqLen)


## classify on test data with NGSclassifier (the default classifier)

result <- MSIseq.classify(mutationNum = test.mutationNum)
```

MSIseq.train — *Build Microsatellite Instability Classification Model with Training Dataset*

Description

This function generate a detector for MSI status based on the mutation information in the `mutationNum` parameter.

Usage

```
MSIseq.train(mutationNum, classification, cancerType = NULL)
```

Arguments

mutationNum	A data frame output from Compute.input.variables, which containing 9 variables: T.sns, S.sns, T.ind, S.ind, T, S, Ratio.sns, Ratio.ind, Ratio.
classification	A data frame with two columns: Tumor_Sample_Barcode (tumor ID) and the corresponding MSI_status. Check NGStrainclass for detail.
cancerType	A data frame with two columns: Tumor_Sample_Barcode (tumor ID) and the corresponding cancer_type. Check NGStraintype for detail.

Details

This function builds and evaluates a decision tree model from mutationNum.

Value

A Weka_classifier object: a decision tree model built with the 'RWeka' function J48()

Author(s)

Mini Huang

References

Kurt Hornik, Christian Buchta, Achim Zeileis (2009) Open-Source Machine Learning: R Meets Weka. Computational Statistics, 24(2), 225-232.

See Also

MSIseq.classify, Compute.input.variables

Examples

```
## load sample data (train.mutationNum, NGStraintype,
## NGStrainclass)

data(train.mutationNum)
data(NGStrainclass)
data(NGStraintype)

## create NGSclassifier with traindata
## note that this is a built-in classifier, which can be directly used
## if you do not have your own training data to create a classifier

NGSclassifier<-MSIseq.train(mutationNum = train.mutationNum,
  classification=NGStrainclass, cancerType=NGStraintype)
```

NGStestdata *Mutation Annotation Data with Test Set Samples*

Description

This is a data frame based on the TCGA "mutation annotation file" format.

Usage

```
data(NGStestdata)
```

Format

A data frame with 72,111 observations on the following 5 variables:

`Chrom` a character vector that indicates the chromosome identifier without the "chr" prefix: "1", "2", ..., "22", "X", "Y"

`Start_Position` a numeric vector

`End_Position` a numeric vector

`Variant_Type` a character vector that indicates the type of variant; legal values are "SNP", "INS" and "DEL". Other values will cause the `Compute.input.variables` to generate an error

`Tumor_Sample_Barcode` the sample ID as a character vector

Any other columns are ignored.

Details

This is sample input data for the `Compute.input.variables` function.

Source

https://tcga-data.nci.nih.gov/tcga/

References

Mutation Annotation File Format https://wiki.nci.nih.gov/display/TCGA/File+Format+Specifications

NGStestseqLen *Capture Sequence Length for Test Set Samples*

Description

This is a data frame containing the capture sequence lengths for each sample in NGStestdata.

Usage

```
data(NGStestseqLen)
```

Format

A data frame with 186 observations on the following 2 variables:

`Tumor_Sample_Barcode` the sample ID as a character vector

`Sequence_Length` a numeric vector indicating capture sequence length for each sample; the length unit is Mb.

Details

This is the sample data for the `Compute.input.variables` function input.

Source

https://tcga-data.nci.nih.gov/tcga/

NGStesttype *Cancer Types for the TCGA Test Set Samples*

Description

This is a data frame containing the cancer types for each sample in `test.mutationNum`.

Usage

```
data(NGStesttype)
```

Format

A data frame with 426 observations on the following 2 variables:

`Tumor_Sample_Barcode` the sample ID as a character vector

`cancer_type` a factor indicating cancer types for each sample

Details

This is the sample data for MSIseq.classify function input.

Source

https://tcga-data.nci.nih.gov/tcga/

NGStrainclass *MSI Status Classification for the TCGA Training Set Samples*

Description

This is a data frame containing the MSI status for each sample in train.mutationNum.

Usage

```
data(NGStrainclass)
```

Format

A data frame with 426 observations on the following 2 variables:

`Tumor_Sample_Barcode` the sample ID as a character vector

`MSI_status` a factor indicating MSI status for each sample, containing two levels: "MSI-H" and "Non-MSI-H". Other values will cause MSIseq.train to generate an error.

Details

This is the sample data for MSIseq.train function input.

Source

https://tcga-data.nci.nih.gov/tcga/

NGStraindata *Mutation Annotation Data with Training Set Samples*

Description

This variable has the same format as NGStestdata.

Usage

```
data(NGStraindata)
```

Format

A data frame with 152,842 observations on the following 5 variables:

`Chrom` a character vector that indicates the chromosome identifier without the "chr" prefix: "1", "2", ..., "22", "X", "Y"

`Start_Position` a numeric vector

`End_Position` a numeric vector

`Variant_Type` a character vector that indicates the type of variant; legal values are "SNP", "INS" and "DEL". Other values will cause the `Compute.input.variables` to generate an error

`Tumor_Sample_Barcode` the sample ID as a character vector

Details

This is sample input data for the `Compute.input.variables` function.

Source

https://tcga-data.nci.nih.gov/tcga/

References

Mutation Annotation File Format https://wiki.nci.nih.gov/display/TCGA/File+Format+Specifications

NGStrainseqLen *Capture Sequence Length for Training Set Samples*

Description

This is a data frame containing the capture sequence lengths for each sample in NGStraindata.

Usage

```
data(NGStrainseqLen)
```

Format

A data frame with 426 observations on the following 2 variables:

`Tumor_Sample_Barcode` the sample ID as a character vector

`Sequence_Length` a numeric vector indicating capture sequence length for each sample; the length unit is Mb.

Details

This is the sample data for the Compute.input.variables function input.

Source

https://tcga-data.nci.nih.gov/tcga/

NGStraintype *Cancer Types for the TCGA Training Set Samples*

Description

This is a data frame containing the cancer types for each sample in train.mutationNum.

Usage

```
data(NGStraintype)
```

Format

A data frame with 426 observations on the following 2 variables:

`Tumor_Sample_Barcode` the sample ID as a character vector

`cancer_type` a factor indicating cancer types for each sample

Details

This is the sample data for MSIseq.train function input.

Source

https://tcga-data.nci.nih.gov/tcga/

---

read.fasta *Read Fasta File*

Description

This function reads in a file with multiple fasta records and return a list.

Usage

```
read.fasta(file.name)
```

Arguments

file.name	The path to a fasta file with multiple fasta records.

Details

This function reads in a file with multiple fasta records and return a list of each chromosome and their sequence context.

Value

A list of each chromosome and their sequence context.

Author(s)

Mini Huang

See Also

find.mono.repeats

test.mutationNum *Table of Mutation Numbers in Test Set Samples*

Description

This is a data frame containing the counts of 9 mutation types for each sample in NGStestdata. It was generated from Compute.input.variables

Usage

```
data(test.mutationNum)
```

Format

A data frame with 165 observations on the following 9 variables:

T.sns total count of SNSs/Mb

S.sns count of SNSs in simple sequence repeats/Mb

T.ind total count of indels/Mb

S.ind count of indels in simple sequence repeats/Mb

T total mutation count/Mb

S mutation count in simple sequence repeats/Mb

ratio.sns S.sns/T.sns

ratio.ind S.ind/T.ind

ratio S/T

Details

This is the sample data for MSIseq.classify function input.

Source

https://tcga-data.nci.nih.gov/tcga/

train.mutationNum *Table of Mutation Numbers in Train Set Samples*

Description

This is a data frame containing the counts of 9 mutation types for each sample in NGStraindata. It was generated from Compute.input.variables

Usage

```
data(train.mutationNum)
```

Format

A data frame with 361 observations on the following 9 variables:

T.sns total count of SNSs/Mb

S.sns count of SNSs in simple sequence repeats/Mb

T.ind total count of indels/Mb

S.ind count of indels in simple sequence repeats/Mb

T total mutation count/Mb

S mutation count in simple sequence repeats/Mb

ratio.sns S.sns/T.sns

ratio.ind S.ind/T.ind

ratio S/T

Details

This is the sample data for MSIseq.train function input.

Source

https://tcga-data.nci.nih.gov/tcga/

Index