Package ‘ampir’

June 29, 2021

Type Package

Title Predict Antimicrobial Peptides

Version 1.1.0

Date 2021-06-29

Description
A toolkit to predict antimicrobial peptides from protein sequences on a genome-wide scale. It incorporates two support vector machine models ("precursor" and "mature") trained on publicly available antimicrobial peptide data using calculated physico-chemical and compositional sequence properties described in Meher et al. (2017) <doi:10.1038/srep42362>. In order to support genome-wide analyses, these models are designed to accept any type of protein as input and calculation of compositional properties has been optimised for high-throughput use. For best results it is important to select the model that accurately represents your sequence type: for full length proteins, it is recommended to use the default "precursor" model. The alternative, "mature", model is best suited for mature peptide sequences that represent the final antimicrobial peptide sequence after post-translational processing. For details see Fingerhut et al. (2020) <doi:10.1093/bioinformatics/btaa653>. The 'ampir' package is also available via a Shiny based GUI at <https://ampir.marine-omics.net/>.

URL https://github.com/Legana/ampir

License GPL-2

Encoding UTF-8

Depends R (>= 3.5.0)

Imports Peptides, caret (>= 6.0.0), kernlab, Rcpp, parallel

RoxygenNote 7.1.1

Suggests testthat (>= 3.0.0), knitr, rmarkdown, e1071

VignetteBuilder knitr

LinkingTo Rcpp

Config/testthat/edition 3
aaseq_is_valid

Check protein sequences for non-standard amino acids

Description

Any proteins that contains an amino acid that is not one of the 20 standard amino acids is flagged as invalid

Usage

aaseq_is_valid(seq)

Arguments

seq A vector of protein sequences
Value
A logical vector where TRUE indicates a valid protein sequence and FALSE indicates a sequence with invalid amino acids

Description
This function calculates set physicochemical and compositional features from protein sequences in preparation for supervised model learning

Usage
`calculate_features(df, min_len = 10)`

Arguments
- `df` A dataframe which contains protein sequence names as the first column and amino acid sequence as the second column
- `min_len` Minimum length sequence for which features can be calculated. It is an error to provide sequences with length shorter than this

Value
A dataframe containing numerical values related to the protein features of each given protein

Note
This function depends on the Peptides package

References

Examples
```r
my_protein_df <- read_faa(system.file("extdata/bat_protein.fasta", package = "ampir"))
calculate_features(my_protein_df)
## Output (showing the first six output columns)
# seq_name Amphiphilicity Hydrophobicity pI Mw Charge ....
# [1] G1P6H5_MYOLU 0.4145847 0.4373494 8.501312 9013.757 4.53015 ....
```
calc_amphiphilicity

Calculate amphiphilicity (or hydrophobic moment)

Usage

calc_amphiphilicity(seq)

Arguments

seq A protein sequence

References


calc_hydrophobicity

Calculate the hydrophobicity

Usage

calc_hydrophobicity(seq)

Arguments

seq A protein sequence

References

**calc_mw**

*Calculate the molecular weight*

**Description**

Calculate the molecular weight

**Usage**

`calc_mw(seq)`

**Arguments**

- `seq` A protein sequence

**References**


---

**calc_net_charge**

*Calculate the net charge*

**Description**

Calculate the net charge

**Usage**

`calc_net_charge(seq)`

**Arguments**

- `seq` A protein sequence

**References**

**calc_pI**  
*Calculate the isoelectric point (pI)*

**Description**
Calculate the isoelectric point (pI)

**Usage**
calc_pI(seq)

**Arguments**
- **seq**: pI

**References**

**calc_pseudo_comp**  
*Calculate the pseudo amino acid composition*

**Description**
This function is adapted from the extractPAAC function from the protr package (https://github.com/nanxstats/protr)

**Usage**
calc_pseudo_comp(seq, lambda_min = 4, lambda_max = 19)

**Arguments**
- **seq**: A vector of protein sequences as character strings
- **lambda_min**: Minimum allowable lambda. It is an error to provide a protein sequence shorter than lambda_min+1
- **lambda_max**: For each sequence lambda will be set to one less than the sequence length or lambda_max, whichever is smaller

**References**
chunk_rows

Determine row breakpoints for dividing a dataset into chunks for parallel processing

Usage

chunk_rows(nrows, n_cores)

Arguments

nrows The number of rows in the dataset to be chunked
n_cores The number of cores that will be used for parallel processing

Value

A list of integer vectors consisting of the rows in each chunk

df_to_faa

Save a dataframe in FASTA format

Description

This function writes a dataframe out as a FASTA format file

Usage

df_to_faa(df, file = "")

Arguments

df a dataframe containing two columns: the sequence name and amino acid sequence itself
file file path to save the named file to

Value

A FASTA file where protein sequences are represented in two lines: The protein name preceded by a greater than symbol, and a new second line that contains the protein sequence
predict_amps

Examples

my_protein <- read_faa(system.file("extdata/bat_protein.fasta", package = "ampir"))

# Write a dataframe to a FASTA file
df_to_faa(my_protein, tempfile("my_protein.fasta", tempdir()))

predict_amps

Predict the antimicrobial peptide probability of a protein

Description

This function predicts the probability of a protein to be an antimicrobial peptide

Usage

predict_amps(faa_df, min_len = 5, n_cores = 1, model = "precursor")

Arguments

- **faa_df**: A dataframe obtained from read_faa containing two columns: the sequence name (seq_name) and amino acid sequence (seq_aa)
- **min_len**: The minimum protein length for which predictions will be generated
- **n_cores**: On multicore machines split the task across this many processors. This option does not work on Windows
- **model**: Either a string with the name of a built-in model (mature, precursor), OR, A train object suitable for passing to the predict.train function in the caret package. If omitted the default model will be used.

Value

The original input data.frame with a new column added called prob_AMP with the probability of that sequence to be an antimicrobial peptide. Any sequences that are too short or which contain invalid amin acids will have NA in this column

Examples

my_bat_faa_df <- read_faa(system.file("extdata/bat_protein.fasta", package = "ampir"))
predict_amps(my_bat_faa_df)
# seq_name   prob_AMP
# [1] G1P6H5_MYOLU 0.9723796
**read_faa**

*Read FASTA amino acids file into a dataframe*

**Description**

This function reads a FASTA amino acids file into a dataframe

**Usage**

```r
read_faa(file = NULL)
```

**Arguments**

- `file`: file path to the FASTA format file containing the protein sequences

**Value**

Dataframe containing the sequence name (seq_name) and sequence (seq_aa) columns

**Note**

This function was adapted from `read.fasta.R` by Jinlong Zhang (jinlongzhang01@gmail.com) for the phylotools package (http://github.com/helixcn/phylotools)

**Examples**

```r
read_faa(system.file("extdata/bat_protein.fasta", package = "ampir"))
## Output
# seq_name seq_aa
# [1] G1P6H5_MYOLU MALTVRIQAACLLLLLASLTSYSL....
```

**remove_nonstandard_aa**

*Remove non standard amino acids from protein sequences*

**Description**

This function removes anything that is not one of the 20 standard amino acids in protein sequences

**Usage**

```r
remove_nonstandard_aa(df)
```

**Arguments**

- `df`: A dataframe which contains protein sequence names as the first column and amino acid sequence as the second column
Value
a dataframe like the input dataframe but with removed proteins that contained non standard amino acids

Examples

```r
non_standard_df <- readRDS(system.file("extdata/non_standard_df.rds", package = "ampir"))

# non_standard_df
# seq_name  seq_aa
# [1] G1P6H5_MYOLU MALTVRIQAACLLLLLLASLTSYSLLLSQTQLADLQTQ....
# [2] fake_sequence MKVTHEUSYR$GXMBIJDG=M80-%

remove_nonstandard_aa(non_standard_df)
# seq_name  seq_aa
# [1] G1P6H5_MYOLU MALTVRIQAACLLLLLLASLTSYSLLLSQTQLADLQTQ....

remove_stop_codon
Remove stop codon at end of sequence

Description
Stop codons at the end of the amino acid sequences are removed

Usage
remove_stop_codon(faa_df)

Arguments
faa_df A dataframe containing two columns: the sequence name and amino acid sequence

Value
The input dataframe without the stop codons at the end of sequences
Index

aaseq_is_valid, 2

calc_amphiphilicity, 4
calc_hydrophobicity, 4
calc_mw, 5
calc_net_charge, 5
calc_pl, 6
calc_pseudo_comp, 6
calculate_features, 3
chunk_rows, 7

df_to_faa, 7

predict_amps, 8

read_faa, 9
remove_nonstandard_aa, 9
remove_stop_codon, 10