Package: abasequence (via r-universe)

October 31, 2024

Title Coding 'ABA' Patterns for Sequence Data

Version 0.1.0

Description Provides a suite of functions for analyzing sequences of events. Users can generate and code sequences based on predefined rules, with a special focus on the identification of sequences coded as 'ABA' (when one element appears, followed by a different one, and then followed by the first). Additionally, the package offers the ability to calculate the length of consecutive 'ABA'-coded sequences sharing common elements. The methods implemented in this package are based on the work by Ziembowicz, K., Rychwalska, A., & Nowak, A. (2022). <doi:10.1177/10464964221118674>.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author Andrew Pilny [aut, cre] (<https://orcid.org/0000-0001-6603-5490>)

Maintainer Andrew Pilny <andy.pilny@uky.edu>

Date/Publication 2023-07-14 13:20:02 UTC

Repository https://peerreview269.r-universe.dev

RemoteUrl https://github.com/cran/abasequence

RemoteRef HEAD

RemoteSha d43c810d5de725375e2f056050ad19b71a8a9fe8

Contents

`count_events` *Count the Number of Occurrences of Each Event in a Sequence*

Description

This function counts the number of occurrences of each unique event in a sequence. The result is a dataframe with two columns: ID and Frequency.

Usage

```
count_events(event_vector)
```

Arguments

`event_vector`	A numeric vector representing a sequence of events.

Value

A dataframe with two columns: ID and Frequency, showing the number of occurrences of each event.

Examples

```
speaker_no <- c(3, 2, 3, 1, 4, 2, 4, 1, 4, 3, 2, 3)
count_events(speaker_no)
```

`create_is_aba` *Create a Dummy Variable Indicating Whether a Code Represents 'ABA' (1) or not (0).*

Description

This function creates a dummy variable indicating whether a code represents 'ABA'.

Usage

```
create_is_aba(codes_df)
```

Arguments

`codes_df`	A dataframe of binary codes generated by the generate_codes function.

Value

A dataframe of codes with an additional column indicating whether the code represents 'ABA'.

Examples

```
speaker_no <- c(3, 2, 3, 1, 4, 2, 4, 1, 4, 3, 2, 3)
sequences_df <- generate_sequences(speaker_no, 3)
codes_df <- generate_codes(sequences_df)
create_is_aba(codes_df)
```

---

generate_codes *Generate Codes for Sequences Based on Certain Rules*

Description

This function generates one of four possible codes for sequences: AAA, ABA, ABB, ABC.

Usage

```
generate_codes(sequences)
```

Arguments

sequences	A dataframe containing the input sequences.

Value

A dataframe of sequences with their corresponding codes.

Examples

```
speaker_no <- c(3, 2, 3, 1, 4, 2, 4, 1, 4, 3, 2, 3)
sequences_df <- generate_sequences(speaker_no, 3)
generate_codes(sequences_df)
```

---

generate_length_aba *Generate Length of Consecutive 'ABA'-Coded Sequences*

Description

This function calculates the length of consecutive 'ABA'-coded sequences that share common elements in their ID. It assigns NA to non-'ABA' codes.

Usage

```
generate_length_aba(codes_df)
```

Arguments

codes_df	A dataframe of codes generated by the generate_codes function and processed by the create_is_aba function.

Value

A dataframe of codes with an additional column representing the length of 'ABA' sequences.

Examples

```
speaker_no <- c(3, 2, 3, 1, 4, 2, 4, 1, 4, 3, 2, 3)
sequences <- generate_sequences(speaker_no, 3)
codes <- generate_codes(sequences)
aba <- create_is_aba(codes)
length_aba <- generate_length_aba(aba)
```

---

generate_sequences *Generate Sequences of a Given Length from a Numeric Vector*

Description

This function generates sequences of a given length from a numeric vector.

Usage

```
generate_sequences(event_vector, sequence_length)
```

Arguments

event_vector	A numeric vector representing a sequence of events.
sequence_length	An integer representing the length of sequences to generate. Currenlty only supported with sequence lengths of 3

Value

A dataframe containing the sequences and their ID.

Examples

```
speaker_no <- c(3, 2, 3, 1, 4, 2, 4, 1, 4, 3, 2, 3)
generate_sequences(speaker_no, 3)
```

Index