Package: SoundexBR (via r-universe)

September 7, 2024

Type Package

Title Phonetic-Coding for Portuguese

Version 1.2

Date/Publication 2014-12-12

Author Daniel Marcelino

Maintainer Daniel Marcelino <dmarcelino@live.com>

Description The SoundexBR package provides an algorithm for decoding names into phonetic codes, as pronounced in Portuguese. The goal is for homophones to be encoded to the same representation so that they can be matched despite minor differences in spelling. The algorithm mainly encodes consonants; a vowel will not be encoded unless it is the first letter. The soundex code resultant consists of a four digits long string composed by one letter followed by three numerical digits: the letter is the first letter of the name, and the digits encode the remaining consonants.

Suggests SciencesPo, stringr

LazyData yes

License GPL (>= 2)

Encoding latin1

Depends R (>= 3.0),stats,utils,graphics,grDevices

Enhances RecordLinkage

BugReports http://github.com/danielmarcelino/soundexBR

ByteCompile TRUE

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

RemoteUrl https://github.com/dmarcelinobr/soundexbr

RemoteRef HEAD

RemoteSha e77d23d6bb40991882342bc4abfa4b6c9b405fe5

Contents

accent 2
ascii.table 3
char2int 3
int2char 4
soundexBR 4

Index **8**

`accent` *Get rid of Accent Marks*

Description

Replace lower and upper case accented letters with their counterpart without diacritical marks.

Usage

```
accent(x)
```

Arguments

`x`	is a data object which contains diacritical marks.

Details

This function can replace a variety of common marks, but not all of them. It is designed to be expanded on demand.

Value

a vector with same length of `x` without diacritic.

Author(s)

Daniel Marcelino <dmarcelino@live.com>.

See Also

`ascii.table`.

Examples

```
x <- 'ThÃ©rÃ¨se, ne me jugez pas de moitiÃ© si vous ne me connaissez pas du tout.'
accent(x)
```

ascii.table *ASCII Characters Table*

Description

To detect ASCII characters, we may need to specify them literally. This function helps identifying what character is in ascii format and what is not.

Usage

```
ascii.table(x)
```

Arguments

x	A string whose characters is to be checked against.

Value

Returns TRUE if ASCII character and FALSE otherwise.

Author(s)

Daniel Marcelino, <dmarcelino@live.com>

char2int *Character to Integer*

Description

Declares characters to integer.

Usage

```
char2int(x)
```

Arguments

x	A vector consisting of characters.

See Also

int2char, iconv.

Examples

```
char2int("This should be whole numbers")
```

int2char *Interger to Character*

Description

Declares integer inputs as UTF-8 output.

Usage

```
int2char(x)
```

Arguments

x	A vector consisting of whole numbers.

See Also

char2int, iconv.

Examples

```
int2char(c(84,104,105,115,32,115,104,111,117,108,
100, 32, 98, 101, 32, 119, 104, 111, 108, 101,
32, 110, 117, 109, 98, 101, 114, 115))
```

soundexBR *Phonetic-Coding For Portuguese*

Description

The soundexBR function is an algorithm for decoding names into phonetic codes, as pronounced in Portuguese. Soundex coding can be useful to identify 'close' matches which typically fail due to variant spellings of names. For instance, both "Clair" and "Claire" return the same string "C460", but the slightly different spellings of these names is enough to cause a deterministic linkage to fail when comparing the actual names. Soundex does not help when the variants do not sound alike, or start with different letters.

Usage

```
soundexBR(term, BR=TRUE, useBytes = FALSE)
```

Arguments

term	a list, a vector or a data frame with character strings.
BR	if BR=TRUE, commong mispelled first letters may be replaced
.	
useBytes	if useBytes=TRUE performs byte-wise comparison. The default is set to FALSE, which takes longer, but it may prevent different results depending on character encoding.

Details

The soundexBR may help with identification of names even when they are spelled differently. However, the algorithm does not help when the variants do not sound alike, or start with different letters. For instance, while "Carolina" and "Karolina" both receive a similar code, respectively "C645" and "K645", the first letter differ. Further, this function is only meaningful for characters in the ranges a-z and A-Z. Although, I tried to minimize character encoding issues, there are several non-printable ascii characters and system-dependent characters that may cause the function to print a warning message.

The numerical digits of the code are based on specific consonant sounds and can be computed by using the letters's 'power'. For instance, (1) retain the first letter of the name and drop other occurrences of A, E, I, O, U, Y, H, W. (2) replace consonants with digits as follows (after the first letter):

```
B, F, P, V = 1
C, G, J, K, Q, S, X, Z = 2
D, T = 3
L = 4
M, N = 5
R = 6
```

(3) If two or more letters with the same number are adjacent in the original name (before step 1), only retain the first letter; also two letters with the same number separated by 'h' or 'w' are coded as a single number, whereas such letters separated by a vowel are coded twice, for instance, "Ashcraft" and "Ashcroft" both yield "A261" (the chars 's' and 'c' in the name receive a single number of 2 and not 22 since an 'h' lies in between them). (4) Iterate the previous step until you have one letter and three numbers. If you have too few letters in your word that you cannot assign three numbers, append with zeros until there are three numbers as in "A500". If the resultant code has more than 3 numbers, just retain the first 3 numbers.

Probabilistic Matching: soundexBR may be used as a phonetic identifier to find results that do not match exactly the terms. Searching for names can be difficult as there are often multiple alternative spellings for names. An example is the name Claire. It has two alternatives, Clare and Clair, which are both pronounced the same. Searching for one spelling would not show results for the two others. Fortunately, soundexBR will produce the same code for all three variations, "C460". Therefore, searching names based on the soundex code all three variations will be matched. In the examples below, you can instances of using soundexBR together with the **RecordLinkage** package, by supplying it with a phonetic dictionary for Portuguese records.

Value

A character vector with same dimensions as `term`.

Note

This function was adapted from the US census soundex version. See in http://archives.gov/research/census/soundex.html

Author(s)

Daniel Marcelino <dmarcelino@live.com>

References

Borg, Andreas and Murat Sariyar. (2012) *RecordLinkage: Record Linkage in R,* R package version 0.4-1, http://CRAN.R-project.org/package=RecordLinkage.

Camargo Jr. and Coeli CM. (2000) Reclink: aplicativo para o relacionamento de bases de dados, implementando o método probabilistic record linkage. *Cad. Saúde Pública*, **16(2)**, Rio de Janeiro.

Eastman, Dick *Soundex Calculator*, http://eogn.com/soundex/.

Marcelino, Daniel (2013) *SciencesPo: A Tool Set for Analyzing Political Behaviour Data*, http://dx.doi.org/10.2139/ssrn.2320547.

Paula, Fátima de Lima (2014) *Readmissão Hospitalar de Idosos após Internação por Fratura Proximal do Fêmur no Município do Rio de Janeiro*, Doctoral thesis, Fiocruz.

Examples

```
#### A silly example:
names <- c('Ana Karolina Kuhnen', 'Ana Carolina Kuhnen', 'Ana Karolina','João Souza',
'João Souza', 'Dilma Vana Rousseff', 'Dilma Rousef','Aécio Neves', 'Aecio Neves')

soundexBR(names)

# Example with RecordLinkage:
#Some data:
data1 <- data.frame(list(
fname=c('Ricardo','Maria','Tereza','Pedro','José','Rubens'),
lname=c('Cunha','Andrade','Silva','Soares','Silva','Lima'),
age=c(67,89,78,65,68,67),
birth=c(1945,1923,1934,1947,1944,1945),
date=c(20120907,20120703,20120301,20120805,20121004,20121209)
))

data2<-data.frame( list( fname=c('Maria','Lúcia','Paulo','Marcos','Ricardo','Rubem'),
lname=c('Andrada','Silva','Soares','Pereira','Cunha','Lima'),
age=c(67,88,78,60,67,80),
birth=c(1945,1924,1934,1952,1945,1932),
date=c(20121208,20121103,20120302,20120105,20120907,20121209)
))
```

```
# Must call RecordLinkage package

## Not run: pairs<-compare.linkage(data1, data2,
blockfld=list(c(1,2,4),c(1,2)),
phonetic<-c(1,2), phonfun = soundexBR, strcmp = FALSE,
strcmpfun<-jarowinkler, exclude=FALSE,identity1 = NA,
identity2=NA, n_match <- NA, n_non_match = NA)

print(pairs)

editMatch(pairs)

# To access information in the object:
weights <- epiWeights(pairs, e = 0.01, f = pairs$frequencies)
hist(weights$Wdata, plot = FALSE) # Plot TRUE
getPairs(pairs, max.weight = Inf, min.weight = -Inf)

## End(Not run)
```

Index

∗ **Attributes**
accent, 2
∗ **Encoding**
char2int, 3
int2char, 4
∗ **Soundex**
soundexBR, 4

accent, 2
ascii.table, *2*, 3

char2int, 3, *4*

iconv, *3*, *4*
int2char, *3*, 4

soundexBR, 4