Introduction to the gower package

Mark van der Loo

May 14, 2019

Contents

1 Gower's distance measure

Gower's distance can be used to measure how different two records are. The records may contain combinations of logical, numerical, categorical or text data. The distance is always a number between 0 (identical) and 1 (maximally dissimilar). An easy to read specification of the measure is given in the original paper.

Gower (1971) A general coefficient of similarity and some of its properties. *Biometrics* **27** 857–874.

In short, Gower's distance (or similarity) first computes distances between pairs of variables over two data sets and then combines those distances to a single value per record-pair.

This package modifies Gower's original similarity measure in the following ways.

- In stead of the original similarity S, the package returns the distance $1-S$.
- The original paper does not mention the concept of `NA`. Missing variables are skipped when computing the distance.
- The original paper does not mention character data. These are treated as categorical data.

2 Computing Gower's distance

The function `gower_dist` computes pairwise-distances between records.

```
library(gower)
dat1 <- iris[1:10,]
dat2 <- iris[6:15,]
gower_dist(dat1, dat2)
```

```
 [1] 0.5155844 0.2155844 0.2125541 0.1316017 0.2718615 0.3696970 0.2619048
 [8] 0.2679654 0.3324675 0.5922078
```

If one data frame has less records than the other, the shortest one is recycled over (just like when you're adding two vectors of unequal length)

```
gower_dist(iris[1,], dat1)
```

```
 [1] 0.00000000 0.16111111 0.23333333 0.28055556 0.04722222 0.56388889
 [7] 0.24722222 0.11388889 0.30833333 0.30555556
```

It is possible to control how columns from the two data sets are paired for comparison using the `pair_x` and `pair_y` arguments. This comes in handy when similar colums have different names accross datasets. By default, columns with matching names are paired. The behaviour is somewhat similar to that of base R's `merge` in that respect.

```
dat1 <- dat2 <- iris[1:10,]
names(dat2) <- tolower(names(dat2))
gower_dist(dat1, dat2)
# tell gower_dist to match columns 1..5 in dat1 with column 1..5 in dat2
gower_dist(dat1, dat2, pair_y=1:5)
```

```
 [1] 0 0 0 0 0 0 0 0 0 0
```

It is also possible to explicitly ignore case when matching columns by name.

```
gower_dist(dat1, dat2, ignore_case=TRUE)
```

```
 [1] 0 0 0 0 0 0 0 0 0 0
```

3 Computing the top-n best matches

The function `gower_topn` returns a list with two arrays.

```
dat1 <- iris[1:10,]
L <- gower_topn(x=dat1, y=iris, n=3)
L
```

```
$index
      row
topn   [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
  [1,]    1    2    3    4    5    6    7    8    9    10
  [2,]   18   35   48   48   38   17   48   40   39    35
  [3,]   28   26   30   31    1   45   12   50   43    31

$distance
      row
topn          [,1]       [,2]        [,3]       [,4]       [,5]       [,6]
  [1,] 0.000000000 0.00000000 0.000000000 0.00000000 0.00000000 0.00000000
  [2,] 0.008333333 0.01178161 0.009003831 0.01178161 0.01388889 0.01379310
  [3,] 0.009003831 0.01245211 0.010344828 0.01455939 0.01388889 0.03189655
      row
topn       [,7]        [,8]       [,9]       [,10]
  [1,] 0.000000 0.000000000 0.00000000 0.000000000
  [2,] 0.025000 0.005555556 0.01178161 0.008333333
  [3,] 0.026341 0.011781609 0.02844828 0.017337165
```

The first array is called `index`. Each column corresponds to one row of `x`. The entries of each column index the top n best matches of that row in `x` with rows in `y`. In this example, the best match of the first row of `dat1` is record number 1 from `iris` (this should be obvious, since they are the same record). The second best match is record number 18 from `iris`.

The second array is called `distance` and it contains the corresponding distances.

4 Using weights

Gower's distance is computed as an average over differences between variables. By setting `weights` you can compute the distance as a weighted average.

```
gower_dist(women[1,], women)
```

```
 [1] 0.00000000 0.05612245 0.12244898 0.18877551 0.25510204 0.32142857
 [7] 0.38775510 0.45408163 0.53061224 0.59693878 0.67346939 0.75000000
[13] 0.82653061 0.91326531 1.00000000
```

```
gower_dist(women[1,], women, weights=c(2,3))
```

```
 [1] 0.00000000 0.05306122 0.11836735 0.18367347 0.24897959 0.31428571
 [7] 0.37959184 0.44489796 0.52244898 0.58775510 0.66530612 0.74285714
[13] 0.82040816 0.91020408 1.00000000
```

5 Parallelization, memory usage

The underlying algorithm is implemented in C and parallelized using OpenMP. OpenMP is available on most systems that can run R. Please see this section of the writing R extensions manual for up-to-details details on which systems are supported. At the time of writing (spring 2019), OSX is the only system not supporting OpenMP out of the box. You can still make it work by installing the gcc toolchain and compiling the package (and R).

If OpenMP is not supported, the package will still work but the core algorithms will not be parallelized.

This implementation makes no copies of the data in memory. When computing `gower_dist`, two double precision arrays of size `max(nrow(x),nrow(y))` are kept in memory to store intermediate results. When computing the top-n matches, for k cores, $k+2$ double precision arrays of length `nrow(y)` are created to store intermediate results at C level.