kRecentMonths

Mark White | markhwhiteii@gmail.com

5/17/2017

This function is derivative of the `newAbbrvReport` function; it uses the same `newAbbrvCounts` data that we feed into it. Instead of returning words that have never been used before the current month (along with context and BioSense ID), this function returns words that *have* been used in the current month and before, but just not in the most recent *k* previous months.

Preparing to use the `kRecentMonths` function

The function requires data that is structured as follows:
1. The first column is a vector of words used in chief complaint or admit reason fields.
2. Each subsequent column is a vector of numbers that represent the word count for that word in a given month.
3. Each subsequent column should represent a month, and these months *must be in chronological order from left to right*.
4. There are no more columns in the data file than the ones described here.

Here is what an example of what this data frame might look like:

word	n_spt16	n_oct16	n_nov16	n_dec16	n_jan17	n_feb17	n_mar17	n_apr17
pain	23408	23082	21937	21566	23129	21643	23788	22690
pt	7964	8346	7860	7064	7664	8113	9018	8809
left	5452	5154	4756	4469	4501	4412	5411	5463
unspecified	4859	4684	4258	4408	4880	4522	4676	4141
chest	4598	4601	4313	4569	4999	4775	5090	4675
injury	4558	4623	4020	3609	3483	3597	4021	4174

Note that the first column are all the words, and each subsequent column represents counts for a given month, and these columns are in chronological order. Lastly, note that this data frame contains these columns and *only* these columns. This is the default structure of the `newAbbrvCounts` file that is described in the `newAbbrvReport` documentation documents. Once these data have been loaded in, one simply runs the `kRecentMonths.R` document to load the function, and then the function is ready to be used. There are two arguments:
1. k = the number of recent months one wants to say the word hasn't been used in. For example, if we were looking for a word that has been used in the current month and before, but not in the past 4 months, one would specify `k = 4`.
2. dat = the data frame, with the structure described above, one wants to use.

Using the `kRecentMonths` function

As an example, let's say that we want to know words that were used in April 2017 that appeared for the first time *in three months*. That is, the word appeared in April, did not appear in January, February, or March, and did appear before January. One would specify:

```
result <- kRecentMonths(k=3, dat=data)
```

Let's look at the first six rows:

```
kable(head(result))
```

word	n_spt16	n_oct16	n_nov16	n_dec16	n_jan17	n_feb17	n_mar17	n_apr17
gardening	3	0	0	0	0	0	0	6
mayo	7	3	3	2	0	0	0	5
rsd	0	0	1	0	0	0	0	5
backyard	3	2	0	0	0	0	0	3
spraying	3	1	2	1	0	0	0	3
weeds	3	2	1	0	0	0	0	3

This function outputs the words in descending order from most to least used in the most recent month (in this case, April of 2017). As we can see, "gardening" appears for the first time since September of 2016. This list can get very long. How many words did we return for this search?

```
nrow(result)

## [1] 1327
```

1,327 is a lot of words to parse through! Most of these are just used *once*, as well:

```
table(result$n_apr17)

## 
##    1    2    3    5    6 
## 1111  172   41    2    1
```

So one could specify only looking at the ones used more than once by specifying wanting rows where the *n* for April 2017 was greater than 1:

```
result[which(result$n_apr17>1),]
```

An equivalent argument for this using the `subset()` function instead of brackets would be:

```
subset(result, n_apr17>1)

## # A tibble: 216 × 9
##           word n_spt16 n_oct16 n_nov16 n_dec16 n_jan17 n_feb17 n_mar17
##          <chr>   <int>   <int>   <int>   <int>   <int>   <int>   <int>
## 1    gardening       3       0       0       0       0       0       0
## 2          mayo       7        3        3        2        0        0        0
## 3           rsd       0        0        1        0        0        0        0
## 4      backyard       3        2        0        0        0        0        0
## 5      spraying       3        1        2        1        0        0        0
## 6         weeds       3        2        1        0        0        0        0
## 7        advice       2        2        0        1        0        0        0
## 8   cooperating       2        0        1        0        0        0        0
## 9       dumping       2        0        0        0        0        0        0
## 10         laft       2        0        0        0        0        0        0
## # ... with 206 more rows, and 1 more variables: n_apr17 <int>
```

Explaining the `kRecentMonths` function

I usually create a separate document for going through the code line-by-line, but this function is short enough that I felt the Guide and Explainer documents I usually write could be condensed into one for this function. So let's start at the beginning, going through the code line-by-line.

```
kRecentMonths <- function(k, dat) {
```

This names the function `kRecentMonths` and specifies the two arguments it needs: `k` and `dat`.

```
  if (k + 3 > ncol(dat) | k < 1) {
    cat("The k you selected is not able to be calculated with the current dat
a.")
```

The function doesn't run if `k + 3` is greater than the number of columns in the dataset. One column is the current month, another is the column for the word, and a third column is necessary to say that the word *did* appear before. Let's say we have 8 months worth of data and want to know if a word has been used in the 8th month, not in the previous 8 months, but was used before then. We don't have enough data for this! We are two months short. `k + 3` = 11, which is 2 higher than the total columns we have in the data set. The function also won't run if we select a `k < 1`, obviously. So if the user selects a `k` that falls anywhere into something that doesn't make sense, the function stops and returns the message that the `k` is not appropriate for the current use.

```
  } else {
```

This specifies that if the `k` is acceptable, we move on to doing the actual meat of the function:

```
    rows <- which(
```

We are going to save a number of rows where the word (a) appears in the most recent month, (b) does not appear in the `k` months before that, and (c) did appear before those `k` months. These row numbers are stored in the appropriately named `rows` object.

```
      rowSums(dat[,c(2:(ncol(dat)-k-1))]) > 0 &
```

The first condition is that the words *were* used before the `k` months we specify. So we take the sum of the rows for columns `2` (the first column with counts) up until the max number of rows minus `k` minus `1`. This will select all columns from the beginning of the data we have up until the month right before our `k` recent months start. We want to include the row if the sum of these rows is greater than zero (i.e., the word has been used before).

It is important here that `&` is at the end of this statement; we are about to specify more conditions that these rows must meet.

```
        rowSums(dat[,c((ncol(dat)-k):(ncol(dat)-1))]) == 0 &
```

The next condition is that the words were *not* used in the `k` most recent months. This code sums the rows again: We are selecting the max number of columns (again, the last column is the most recent month's data we are using as a comparison) minus `k` columns up until the max number of columns minus `1`. We want the sum of these rows to equal zero (i.e., the word has not been used in the most recent `k` months preceding the most recent month). Another ampersand, which means we have one last condition to satisfy.

```
        rowSums(dat[,ncol(dat)]) > 0
    )
```

Lastly, we want to see if the last column in the dataset (or the column whose number equals the max number of rows) has a value of greater than zero. If all three of these conditions are met, we save the row number from the dataset into the object `rows`...

```
    dat <- dat[rows,]
    return(dat[order(-dat[,ncol(dat)]),])
  }
}
```

And then we save a new dataset (`dat`) with only the rows (`rows`) that we want. Then we return that dataset, but we specify that we want the rows ordered by their value in the last column of the dataset (i.e., the most recent month), and with the `-` we say that we want this in descending order.