The screenmedR package

Theodoros Diakonidis

21-Jun-2021

Contents

1 Introduction

screenmedR is a package for automatizing the screening of publications in order to minimize the manual work needed to extract the proper publications for your systematic review. It uses unsupervised machine learning algorithms to divide in groups all abstracts in terms of their cosine similarity with a small relevant set of 4 or 5 abstracts which you think they belong to your analysis. The group with the highest cosine similarity can be chosen (in case the difference with the others is big enough) to narrow your search and end up with a much smaller number of abstracts needed for checking. It comes also with extra functions that use mesh terms (Descriptors & Qualifiers) so as to narrow down your search even more. It uses RISmed (Kovalchik, n.d.) and rentrez (Winter 2017) packages to extract the information that needed and tm(Feinerer, Hornik, and Meyer 2008) package for the NLP.

2 Installation

You can download-install the package from github and load the library:

```
#devtools::install_github('thdiakon/screenmedR')
library(screenmedR)
```

3 A case study

This is a simple example of implementation of the code for the screenmed package. The use of it, is to filter the appropriate publications, initially extracted from pubmed using a typical search in order to proceed into a meta-analysis. Collecting the Pubmed Ids (PMID) from a search the program can provide 3 different services.

- Find the most relevant publications for our study, by comparing all the abstracts of the search with those of a small group of publications that we are pretty sure that they belong to the meta-analysis.
- Filter the publications according to the number of common Mesh terms of a small group or a single publication.
- Filter the publications according to a list of specific common Mesh Terms that we are interested in.

3.1 Using screenmed to screen abstracts for a meta-analysis or a systematic review

Here, an example is provided, from a meta-analysis publication [] in order to understand how the program works. One can find more information on the way the program works by reading the above publication.

Initially we applied a search in Pubmed and saved the result in a csv file. We also knew 5 publications that we were quite sure that belong to the study.

```
initial_search <- read.csv(system.file("extdata", "csv-randomized-set.csv", package = "screenmedR"))
initialPMID<-initial_search$PMID
knownPMID<-c("18822428","8276025","16452355","17329276","8346957")
```

3.1.1 The clustering process

Using all that information we apply it to the screenmed function:

```
a_cluster<-screenmed(initialPMID,knownPMID,0.995,2)
```

```
#> Warning in xtfrm.data.frame(x): cannot xtfrm data frames

#> Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

The last term is the number of groups. We chose 2 here as the number of abstracts is quite small, 581. There are a lot of ways of choosing the appropriate numbers of clustering but it will not be discussed further here. Usually for less than 1000 abstracts 2-4 groups would by quite ok. The number before is something that it is useful for the run and sometimes is quite useful for the clustering. It is the number applied to the removesparseterms() function of tm package. It actually removes terms,words that occur very rare (in this case less than 0,5% of the document term matrix). One can play with it between (0,99, 1) to have a better clustering. Bigger difference in Cosine similarity between the two groups in this case.

```
a_cluster$cosine_similarity
```

```
#> [1] 0.7062502 0.4055026
```

These two numbers actually tell us the cosine similarity between the groups of abstracts and the initial (knownPMID) group. The bigger the difference between these two numbers the better the clustering. Generally a difference greater than 0.2 in cosine similarity is quite safe for choosing the one group (the first in this case) and discard the other. It is obvious that if the difference is very small it is very risky to discard it.

You can see the number of abstracts that are clustered using the following command:

```
table(a_cluster$clustering)
```

```
#>
#>   1   2
#> 430  88
```

One can notice that the number of abstracts is smaller than 581. This is due to the fact that not all abstracts are extracted from the RISmed package (missing) so in that case we should add them up to the ones that we see manually.

```
a_cluster$missing_abstracts
```

```
#>  [1] "10420926" "18822428" "8276025"  "8346957"  "17329276" "16452355"
#>  [7] "26049859" "22710761" "26051972" "21501320" "1398249"  "17431678"
#> [13] "12042566" "27641246" "4997825"  "17236880" "17676247" "12113314"
#> [19] "20543721" "1872182"  "20825884" "24190402" "4684368"  "19624374"
#> [25] "16188828" "80053"    "6125663"  "19509348" "4299150"  "12019395"
#> [31] "2735549"  "5365768"  "3513499"  "5243882"  "7815214"  "4746552"
#> [37] "24193957" "15805413" "18041333" "12929303" "3309725"  "3364785"
#> [43] "4890130"  "5433759"  "5225562"  "10592082" "6250107"  "6631614"
#> [49] "6760104"  "7788700"  "470886"   "6382116"  "5990991"  "5524056"
#> [55] "4074569"  "5245334"  "995033"   "5260189"  "3082595"  "1237725"
#> [61] "4717592"  "4403192"  "4015810"
```

So only 88 are out. Is this good enough? Of course not. We can recluster!!!

```
secondPMID<-abstractsofgroup(a_cluster$clustering,1,initialPMID)
```

The abstractsofgroup() function filters the PMIDS of the abstracts that belong to the first group (the one with the bigger cosine similarity). What is left is a second clustering, using the rules of the first:

```
b_cluster<-screenmed(secondPMID,knownPMID,0.992,2)
```

```
#> Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

```
#> Warning in xtfrm.data.frame(x): cannot xtfrm data frames
```

```
b_cluster$cosine_similarity
```

```
#> [1] 0.5742552 0.7535132
```

```
table(b_cluster$clustering)
```

```
#>
#>   1   2
#> 296 134
```

```
table(b_cluster$clustering)
```

```
#>
#>   1   2
#> 296 134
```

3.1.2 Cross check and results

So we ended up with 131+63=194 possible abstracts to check out manually. Not bad. If we can this that we have started from 581. Our colleagues did the job for us. The did the whole check manually and found that the rest are:

```
rest_relevant_pubs<-c("25641242","8835086","23904065","11023168",
                      "10356137","9175947","15930210","8215566","8021760")
```

The cosine similarity result shows that the 2 group should hold our rest_relevant_pubs. Let's check it out:

```
lastPMID<-abstractsofgroup(b_cluster$clustering,2,initialPMID)
intersect(rest_relevant_pubs,lastPMID)
```

```
#> [1] "25641242" "8835086"  "23904065" "11023168" "10356137" "9175947"  "15930210"
#> [8] "8215566"  "8021760"
```

That's correct. Everything is included. Not even needed to check the ones that RISmed could not provide us with abstracts (RISmed is the program that runs behind). The program succeeded to achieve more or less reduction of the manual abstract check to 134/581= 0.23. Not bad at all. Some advice to end up. The process here was a small test. There are meta-analyses that include tens of thousands abstracts. In that case, would be a good idea, to split the initial PMID vector to chunks of 1000 abstract PMIDS and to repeat the process as shown above. One could end up saving a lot of time using the program.

3.2 The mesh functions

Before we proceed let's give some useful information. Let's take one publication from our first group of knownPMID the one with PMID 25641242. If one could check its webpage in pubmed:

https://pubmed.ncbi.nlm.nih.gov/25641242/

he could see that on the left end of the page there is a column with all the mesh terms. If no slash (/) is included the terms are called Descriptors. In case there is a slash the first term is a Descriptor and the second is called Qualifier. The two sets of functions that this package includes are filtering our results, each one in a different way by using those terms that mentioned here.So for example "Blood Pressure / drug effects*" first term has a Descriptor = Blood Pressure and a Qualifier drug effects. It has more than 20 Descriptors and a lot of Qualifiers too.

3.2.1 The clean_mesh() and clean_mesh_bq() functions

Let's say that we would like from our initial check (initialPMID) to restrict more our search. For example we would like to include publications to see which have at least some Descriptors and qualifiers in common with our knownPMID publications. In case of a query that includes less than 300 publications one can use clean_mesh(). If there are more, then we could choose clean_mesh_bq(). Here we have 581, so we use the second:

initialPMID<-initial_search$PMID knownPMID<-c("25641242,""18822428,""8276025,""16452355,""8021760")

```
mesh_clean_bq(initialPMID,knownPMID,11,2)
```

```
#>  [1] 25641242 26890555 18822428 15935918 25039051  8346957 25118721 17329276
#>  [9] 15930210 16452355 19052477 16625228 23904065 17236880
```

Another example, checking the common mesh terms of two publications:

```
a<-c("8276025")
b<-c("8021760")
mesh_clean(a,b,7,2)
```

```
#> [1] "8276025"
```

3.2.2 The mesh_by_name() and mesh_by_name_bq() functions

In the case we need to find publications with specific Mesh terms, Descriptors, Qualifiers, or both, we can use the mesh_by_name() or mesh_by_name_bq() for bigger queries. Eg:

```
Descriptor<-c("Blood Pressure","Dobutamine","Humans","Infant, Newborn")
Qualifier<-c("administration & dosage")
mesh_by_name_bq(initialPMID,Descriptor,Qualifier)
```

```
#> [1] "24100169" "8276025"  "17545944" "9818116"  "15249756" "6700597"
```

The filtering works!!!

Enjoy!

4 Versions of packages compiled for this vignette

R version 4.1.0

package version 1 rentrez 1.2.3 2 XML 3.99.0.6 3 dplyr 1.0.7 4 purrr 0.3.4 5 tm 0.7.8 6 stringr 1.4.0 7 RISmed 2.2 8 tidyr 1.1.3 9 proxy 0.4.26 10 lsa 0.73.2

References

Feinerer, Ingo, Kurt Hornik, and David Meyer. 2008. "Text Mining Infrastructure in r." *Journal of Statistical Software* 25 (5): 1–54. https://www.jstatsoft.org/v25/i05/.

Kovalchik, Stephanie. n.d. "RISmed: A Set of Tools to Extract Bibliographic Content from the National Center for Biotechnology Information (NCBI) Databases, Including PubMed." https://CRAN.R-project.org/package=RISmed.

Winter, David J. 2017. "rentrez: An r Package for the NCBI eUtils API." *The R Journal* 9: 520–26.