Code implementation of the subsampling algorithms.

Introduction

We implemented 3 algorithms, defined "Distance procedure", the "Probability procedure", and the "Uniformity procedure" with the aim of producing sub-samples with specific distributional properties starting from an initial, collection of data which is possibly biased in terms of generalizability to a target population. In the following document we present the R code to implement the algorithms. The specific implementation described here is focused on the Italian PPS study on HAI/AMU prevalence in acute care hospitals but can easily translated to different settings.

The methods try to change the distributional characteristics of a sample by producing a sub-sample whose units are chosen according to some characteristics of interest. Two of these methods, the Distance procedure and the Probability procedure, are aimed at generating a representative sample of a target population, by using reference data at the population level with information on the distribution of relevant characteristics of the observational units. The Uniformity procedure instead generate a sample which is uniform in relation to such characteristics, therefore it does not require population reference data.

This document and all the relative material is available at https://github.com/AD-Papers-Material/SubsamplingMethods.

Input data

The algorithms take as input a database with a statistical unit for each row and the characteristics of interest as columns. Furthermore, a column with an unique ID for each unit is useful for post-hoc checks and mandatory for the Distance procedure. Our algorithms also accept the Quality Score (QS, cfr. Methods) as additional characteristics, but they can be easily modified to remove such feature.

The Probability and the Distance procedures also requires population level reference data, with the same structure. If individual level data is not available at the population, simulated data can be generated given access to the joint distribution of the considered characteristics, ensuring that the simulated dataset is large enough to limit random variation.

Our test case data use acute care hospitals as observational units and hospital size (number of acute care beds) and region of location as characteristics of interest. As a reference, the hospital are also grouped in three hospital size categories; the same categorization is used in the manuscript. We provide simulated sample data for the testing of the procedures at https://github.com/AD-Papers-Material/SubsamplingMethods.

```
# Simulated sample data retaining the real sample characteristics
str(Sample.Data, vec.len = 3)
#> 'data.frame':    143 obs. of  5 variables:
#>  $ Region: chr  "regione piemonte" "regione piemonte" "regione piemonte" ...
#>  $ Beds  : int  183 172 157 179 85 133 189 182 ...
#>  $ QS    : num  101 628 147 109 ...
#>  $ Class : chr  "< 200" "< 200" "< 200" ...
#>  $ Code  : int  1 2 3 4 5 6 7 8 ...

# Official list of Italian acute hospitals, updated to 2016
str(Reference.Data, vec.len = 3)
#> 'data.frame':    963 obs. of  4 variables:
#>  $ Code  : int  10007 10010 10012 10612 10653 10655 10003 10011 ...
#>  $ Beds  : int  258 73 22 105 9 96 337 368 ...
#>  $ Class : chr  "200 - 500" "< 200" "< 200" ...
#>  $ Region: chr  "regione piemonte" "regione piemonte" "regione piemonte" ...
```

General aspects and notation

The procedures at the moment can utilize only **two characteristics** of a sample and these need to be **categorical variables**. Therefore continuous characteristics (specifically, hospital size in this case) are discretized in quantiles by the algorithms before use. The number of quantiles to split continuous features into is an input to the algorithms.
The hospitals are then grouped into *blocks* according to the characteristics of interest: in this study, we used *location/hospital size blocks*, defined by the Italian region ($Region$) and the quantile of number of acute beds ($HSize$) the hospitals fall into. The region and the hospital size quantile allow the definition of a joint discrete probability distribution which is used by the algorithms.
For each block $block_i$ a probability $p_i = Pr(hospital|Region, HSize)$ is defined, either given a sample ($p_{i,sample}$) or the whole country ($p_{i,country}$) which indicate the fraction of hospital in a block over the total.
All the algorithms are costrained by a parameter $N_{required}$ which defines the size of the final sub-sample. As mentioned above, the algorithms may use the Quality Score QS as further discriminant in the selection; to avoid using the QS without code modification is sufficient to assign the same value (a positive number) to all hospitals. Note that lower QS implies better data quality.

Unformity procedure

This procedure sub-samples hospitals trying to obtain an equal proportion of hospitals in every block, by iteratively choosing one hospital from each block. This is the general implementation:

- a candidate list is created from the original sample;
- the hospitals in the list are permuted randomly or ordered in ascending order by QS (the lower the score, the higher the data quality);
- the first hospital is selected;
- all other hospitals belonging to the same block of the selected hospital are removed from the candidate list;
- the process is repeated until there are still available blocks in the candidate list;
- once one hospital from each blocks has been chosen, the hospitals from all the blocks are made available again in the list, apart from those already selected in the sample;
- continue until $N_{required}$ is reached.

For the uniform procedure, we discretized the number of beds only into 4 quantiles since it was less relevant to build a precise discrete probability distribution.

```
uniform.sampling <- function(Input.Sample, n.required, n.quantiles = 4, use.QS = T){

    library(dplyr)
    library(Hmisc)

    # Prepare data by discretizing continuous variables like the number of acute
    # beds and by changing QS to a fixed value if not to be used
    Hospitals <- Input.Sample %>%
        transmute(
            Code, Region,
            Beds = Hmisc::cut2(Beds, g = n.quantiles),
            QS = if (use.QS) QS else 1)

    Selected.hospitals <- c()
    Candidates <- data.frame()

    for (i in 1:n.required) { # Until n.required is reached..

        # If the candidate list is empty, rebuilt it from the non-selected hospitals
        if (nrow(Candidates) == 0) {
```

```
            Candidates <- Hospitals %>%
                # Remove already selected hospitals
                filter(!(Code %in% Selected.hospitals)) %>%
                # Permute order, useful only if QS is not used
                sample_frac() %>%
                # Arrange by QS ascending, best hospitals first
                arrange(QS)
        }

        # Extract the first hospital of the temporary list and add it to the list of
        # selected hospitals
        Extracted.hospital <- Candidates[1,]
        Selected.hospitals <- c(Selected.hospitals, Extracted.hospital$Code)

        # Remove from the temporary list all hospitals in the same location/size block
        # of the extracted hospital
        Candidates <- Candidates %>%
            filter(
                !(Region %in% Extracted.hospital$Region),
                !(Beds %in% Extracted.hospital$Beds)
                )
    }

    # Filter the initial data by the selected hospital codes
    Input.Sample %>% filter(Code %in% Selected.hospitals)
}

## Examples

# Create a subsample of 56 hospitals using the QS
# subsample.uniform(Sample.Data, n.required = 56)

# The same but this time hospitals are chosen randomly
# subsample.uniform(Sample.Data, n.required = 56)
```

Probability procedure

This algorithm uses information from a population level list (Reference Data) to built a discrete probability distribution representative of the target population and then uses it to create a representative sub-sample. The hospitals are selected according to how representative is the block they belong to at the country level. The QS is used to weight such representativeness. The weight of the QS itself can be weighted.

- the blocks are identified in the Reference Data and for each block i the probability $p_{i,country} = Pr(hospital|Region, HSize)$ is computed as the proportion of hospitals in the block over the total;
- these probabilities are assigned to the relative blocks in the sample;
- a score is computer for each hospital j as $score_j = p_{j,country}(1 - scaled.QS_j)^w$ where:
- $p_{j,country}$ is the probability of the block of the hospital j at the country level;
- $scaled.QS_j$ is the QS of the hospital j after that all QS have been rescaled to the range [0,1], with 1 representing the worst quality score and 0 the best. $(1 - scaled.QS_j)$ reweighs the probability of being included of a hospital using the quality of the data;
- w allows scaling the importance of the QS in the selection, with $w = 0$ removing its influence;
- finally, $score_j$ is used to order the hospitals and the first $N_{required}$ get selected. In alternative, the score can be used as a weight for selecting the hospitals by random sampling.

```r
probability.sampling <- function(Input.Sample, Reference.Data, n.required,
    n.quantiles = 10, QS.weight = 1,
    method = c('arrange', 'random')){

    library(dplyr)
    library(Hmisc)
    library(magrittr)
    library(scales)

    method <- match.arg(method)

    # Definition of quantiles in the distribution of number of beds according to
    # reference data
    quantiles <- quantile(Reference.Data$Beds, seq(0, 1, length.out = n.quantiles)) %>%
        round

    P_country <- Reference.Data %>%
        count(Block = Hmisc::cut2(Beds, quantiles) %>% paste('-', Region)) %>%
        mutate(Prob = n / sum(n)) %>%
        with(magrittr::set_names(Prob, Block))

    Selection <- Input.Sample %>%
        mutate(
            # Identification of the country level blocks in the sample
            Block = Hmisc::cut2(Beds, quantiles) %>% paste('-', Region),
            # Association of P_country to the hospital in the sample
            Prob = P_country[Block],
            # Creation of the quality weight after rescaling of the QS
            QS.rescale = 1 - scales::rescale(QS),
            # Definition of the final score
            Score = Prob * QS.rescale^QS.weight
        ) %>%
        filter(!is.na(Score)) # Remove blocks that do not appear in the national list

    if (method == 'arrange') {
        Selection %>%
            arrange(desc(Score)) %>%
            head(n.required)
    } else {
        slice_sample(Selection, n = n.required, weight_by = Score)
    }
}

## Examples

# Create a subsample of 56 hospitals
# subsample.probability(Sample.Data, Reference.Data, n.required = 56)

# Set QS.weight to 0 to not use the QS in the sampling
# subsample.uniform(Sample.Data, Reference.Data, n.required = 56, QS.weight = 0)

# Create a subsample of 56 hospitals with weighted random selection
# subsample.probability(Sample.Data, Reference.Data, n.required = 56, method = 'random')
```

Distance procedure

As with the Probability procedure, the aim is to produce sub-samples that try to reproduce the distributional characteristics of a target population. The advantage of this procedure is that it is particularly appropriate when the original sample is particularly distorted in relation of the characteristics of interest. That is, some blocks are too much underrepresented in the original sample compared to target population and therefore the Probability procedure cannot find enough units in them to reproduce their relative distribution at the population level.
This procedure attempts to solve the problem by oversampling blocks which are similar to the underrepresented ones: if a block cannot provide enough units as required by the expected representativeness at the population level, samples are collected from blocks with similar characteristics. Similarity is defined through ad-hoc distance measures between blocks for each characteristics.
The procedure considers the sample characteristics of interest sequentially, implicitly giving more weight to one or another. Such priority can be passed as an argument.

This algortithm is more complex than the previous two procedure. Here's the general implementation:

- for each i block the expected number of sampled units is computed as $N_{i,expected} = \texttt{Round}(p_{i,country} \times N_{required})$, with $p_{i,country}$ being the target level representativeness of a block as defined above;
- in case $N_{required}$ is not reached, i.e., $(\sum_{j=1}^{n} N_{i,expected}) < N_{required}$, the $N_{i,expected}$ of specific blocks is increased by one unit until $N_{required}$ is achieved. The units are added to blocks whose fractional part of the unrounded $N_{i,expected}$ is lower in absolute value then 0.5 ($|(p_{i,country} \times N_{required}) - N_{i,expected}| < 0.5$), starting by those for which this value is closer to 0.5 (that is, closer to be rounded up);
- blocks are arranged by decreasing $p_{i,country}$ and for each block, $N_{i,sample}$ hospitals are sampled in order of increasing QS (or randomly if the QS is the same for every hospital);
- if $N_{i,sample} < N_{i,expected}$ (i.e., the block is underrepresented), hospitals from similar blocks are *assigned* to it, becoming not available for the other blocks and contributing to the $N_{i,assigned}$ of a block. Similarity is computed via characteristic-specific algorithms (described below), and characteristics are evaluated with a priority chosen by the user; that is either hospital size similarity is considered followed by location similarity or vice-versa. In case of ties, the QS or a random selection is used.
- At this point an initial sub-sample is achieved