Tutorial and PCI example

Institut für Qualitätssicherung und Transparenz im Gesundheitswesen (IQTIG)

2021-09-17

Introduction

This R-package provides an implementation of the statistical methodology for analysing patient survey data as part of quality assurance processes in healthcare. R-functions are provided to obtain point estimate and uncertainty intervals for patient survey based quality indicators (QI). This vignette shows how to use these functions together with an associated `data.frame` to compute two exemplary QIs from the qualitiy assurance domain 'percutaneous coronary interventions and coronary angiography' (PCI). The R package is part of the methodological transparency efforts of the IQTIG and accompanies the detailed statistical description in IQTIG (2018).

```
library(iqtigprm)
```

The Function theta_bayes()

First, we take a look at the core function for evaluating survey data in a Bayesian framework as proposed by the IQTIG methodology. Within the package the most basic buildung block is a function named `theta_bayes`. The function `theta_bayes` computes the posterior distribution for an underlying parameter θ with respect to one quality attribute (in German: 'Merkmal'), given the patient answers y_j regarding the survey items associated with that Merkmal.

Theoretical Background

Suppose, we are interested in a Merkmal consisting of L items within the survey. Each of those items consists of K categories $\{1, \dots, K\}$ ranging from "very poor" to "very good".

Given the underlying competence parameter θ, the likelihood that a patient response y takes one of the K answer categories is modeled through a binomial distribution, i.e.

$$L(y = k|\theta) = \mathbb{P}_{\texttt{Bin}}(x = k-1|\theta, K-1) = \binom{k-1}{K-1}\theta^{k-1}(1-\theta)^{(K-1)-(k-1)},$$

where $\mathbb{P}_{\texttt{Bin}}$ denotes the probability mass function (PMF) of the binomial distribution and k can take any of the category values in $\{1, \dots, K\}$. This likelihood function is referred to as the Hardy-Weinberg binomial model. The intuition behind this model is, that each stepwise improvement in answer category occurs according to a Bernoulli trial with probability θ. With overall K categories, there are at most $K-1$ stepwise category improvements possible. Thus, the patient answer is subject to a Binomial distribution with $K-1$ trials and a probability parameter of θ. For instance, if a patient responds with the third category out of four, the corresponding binomial result is two successes (two categories above the lowest category) out of three trials (at most three category improvements are possible).

The overall likelihood with respect to all patient answers to all considered items is computed by multiplication of the single likelihoods (assuming conditional independence). Conveniently, this again yields a binomial likelihood (up to proportionality) with probability parameter θ. The overall number of trials is given through

the product of the number of patients J times the number of items times the number of categories minus 1, i.e. $J \times L \times (K-1)$. The overall number of successes is given by the sum of the single successes, i.e. let

$$\bar{y} = \sum_{j=1}^{J} \sum_{l=1}^{L} \sum_{k=1}^{K} (k-1) \times \mathbb{I}(y_{jl} = k),$$

where y_{jl} refers to the response of the j-th patient to the l-th item and $\mathbb{I}$ denotes the indicator function. The overall likelihood is then given by

$$L((y_{jl})_{j=1,\ldots,J\, l=1,\ldots,L} | \theta) = \mathbb{P}_{\texttt{Bin}}(x = \bar{y} | \theta, J \times L \times (K-1)).$$

This likelihood function is embedded in an Bayesian framework to infer the competence parameter θ, which assumes a beta-distribution for θ since the beta-distribution is the conjugate prior to the binomial likelihood. Thus, the IQTIG methodology assumes a beta(a, b)-distribution with parameter $a = b = \frac{1}{2}$ as prior for θ, which represents the non-informative Jeffreys prior within this model. Subject to the binomial likelihood, this yields a beta(a^*, b^*)-distribution as posterior for θ, where the parameters a^*, b^* are given by

$$a^* = a + \bar{y} \quad \text{and} \quad b^* = b + (J \times L \times (K-1) - \bar{y}).$$

Thus, the parameters a^*, b^* are updated from the beta-prior parameters (a, b) by adding the number of successes $\bar{y}$ and the number of misses $J \times L \times (K-1) - \bar{y}$ within the binomial likelihood, respectively.

The Function theta_bayes() in Practice

In order to illustrate the mechanics of `theta_bayes` we first simulate some survey data for one fictional Merkmal and one evaluation unit, e.g. one hospital (in German: 'Leistungserbringer'). We assume one associated item ($L = 1$) with $K = 4$ answer categories and that overall $J = 20$ patients of the unit provided survey responses. We also assume, that the unit has an underlying competence parameter $\theta = 0.6$.

Given the number of categories K, we define the possible category values. As notational convention, the function `theta_bayes` as well as the package overall works with response category values on the $[0, 1]$-scale. Thus, the package assumes that the values of an item with K categories are mapped to the K real values $\{0, \frac{1}{K-1}, \ldots, \frac{K-2}{K-1}, 1\}$, rounded to two digits, which yields an equidistant mapping (aside from rounding) of the possible categories to the point scale $[0, 1]$. Hence, the values of "0" and "1" would refer to "very poor" and "very good", respectively. For instance, the answers for an item with four categories would be mapped to the point values $\{0, 0.33, 0.67, 1\}$.

```
## set merkmal specifications
L <- 1
K <- 4

## set LE specification
J <- 20
theta <- 0.6

## define categories
categories <- round(seq(0, 100, length.out = K) / 100, 2)
categories
#> [1] 0.00 0.33 0.67 1.00
```

Subject to θ the binomial model yields the probabilities for the different patient answers categories. Given these probabilities we sample answers for the $J * L$ patients and items.

```
## simulate data for the LE
### category probabilities subject to theta
probs_cat <- dbinom(x = seq(0, K-1), size = K-1, prob = theta)
probs_cat
```

```
#> [1] 0.064 0.288 0.432 0.216
### sample patient answers
set.seed(10000)
y <- sample(categories, prob = probs_cat, size = J * L, replace = TRUE)
table(y)
#> y
#>    0 0.33 0.67    1
#>    2    6    9    3
```

The vector of patient answers `y` serves as data input for the function `theta_bayes`. Additionally required arguments are the number of distinct item categories `nClass`, the assumed beta prior parameters `a` and `b`, as well as the the confidence level `conf_level` which is required for computation of the corresponding credibility interval subject to the resulting posterior for θ.

```
## infer theta from patient answers y according to Bayesian methodology
theta_bayes(y, nClass = K, a = 0.5, b = 0.5, conf_level = 0.95)
#> $a
#> [1] 0.5
#>
#> $b
#> [1] 0.5
#>
#> $astar
#> [1] 33.5
#>
#> $bstar
#> [1] 27.5
#>
#> $hat.post.mean
#> [1] 0.5491803
#>
#> $interval
#>     lower     upper
#> 0.4242473 0.6710568
```

As output the function `theta_bayes` provides a list containing the prior parameters `a` and `b` together with the updated posterior parameters `astar` and `bstar` of the beta posterior. Additionally the output contains the corresponding posterior mean and the credibility interval subject to the provided confidence level. As validation we once manually compute the posterior parameters from `y` as follows.

```
## manual computation
y_bar <- sum(seq(0, K-1)[sapply(y, function(x) which(categories == x))])
y_bar
#> [1] 33

# astar:
0.5 + y_bar
#> [1] 33.5

# bstar:
0.5 + J*L*(K-1) - y_bar
#> [1] 27.5
```

The package in practice: Example from PCI

In the following we illustrate, how the package can be utilized to compute quality indicator results based on patient survey data and formal indicator specifications.

To do so the package provides an artificial survey data set and a subset of indicator specifications from the quality assurance domain PCI. We show the full work flow from data processing to results computation.

Data preprocessing

The package contains an artificial data set consisting of survey 109 patients from two LEs ('Leistungserbringer', i.e. 'hospital'). We take a look at that data.

```
iqtigprm::raw_data_pci %>% glimpse()
#> Rows: 109
#> Columns: 21
#> $ ID             <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ~
#> $ ID_LE          <chr> "LE_A", "LE_A", "LE_A", "LE_A", "LE_A", "LE_A", "LE_A",~
#> $ Fragebogen     <hvn_lbll> 1, 2, 2, 1, 3, 2, 1, 2, 3, 1, 1, 2, 1, 1, 1, 3, 3,~
#> $ FB_Sent        <date> 2018-02-12, 2018-02-13, 2017-12-31, 2018-02-04, 2018-0~
#> $ Gebdatum       <date> 1945-01-07, 1953-05-29, 1942-09-20, 1938-11-06, 1962-0~
#> $ Geschlecht     <hvn_lbll> 2, 2, 2, 2, 2, 2, 1, 2, 1, 1, 2, 2, 1, 1, 2, 1, 2,~
#> $ BMI            <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
#> $ ARermutigtn    <hvn_lbll> 100, 100, 67, 67, 100, 67, 67, 100, 100, -98, 100,~
#> $ ARernstn       <hvn_lbll> 100, 100, 100, 100, 100, 100, 100, 100, 100, -98, ~
#> $ ARrespektn     <hvn_lbll> 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, ~
#> $ ARgelegenheitn <hvn_lbll> -90, 100, 100, 100, -90, 100, 67, 100, -99, -90, 1~
#> $ ARInfverstn    <hvn_lbll> 100, 33, 67, 100, 100, 67, 67, 67, -99, 100, 100, ~
#> $ ARfachwortn    <hvn_lbll> 100, 33, 33, 100, 100, 100, 33, 100, -99, -98, 100~
#> $ ARdeutschn     <hvn_lbll> 100, 100, 100, 100, 100, 100, 100, 100, -99, 100, ~
#> $ ARgesprochenn  <hvn_lbll> 100, 100, 33, 100, 100, 100, 67, 100, -99, -98, 10~
#> $ PAvoranginan   <hvn_lbll> 0, 100, -99, 0, NA, -98, 100, 100, NA, 0, 0, 100, ~
#> $ Anginaruhen    <hvn_lbll> -96, 100, -99, -96, NA, -96, 100, 0, NA, -96, -97,~
#> $ Anginaleichtn  <hvn_lbll> -96, 0, -99, -96, NA, -96, 0, 100, NA, -96, -97, 0~
#> $ PAvorbeeintrn  <hvn_lbll> -96, 75, -99, -96, NA, -96, 50, 50, NA, -96, -97, ~
#> $ Anginaschwern  <hvn_lbll> -96, 100, -99, -96, NA, -96, 100, 100, NA, -96, -9~
#> $ Anginaaussergn <hvn_lbll> -96, 100, -99, -96, NA, -96, 50, 100, NA, -96, -97~
```

The raw survey data contains information on questionnaire answers from the surveyed patients. Each patient represents one row in the data set. It contains a data entry `ID`, the ID of the Leistungserbringer `ID_LE`, some patient characteristics such as birth date `Gebdatum` and sex `Geschlecht` and all survey answers, e.g. the answers to the questionnaire items[1] `ARermutigtn` ("Doctors encouraged me to ask questions during a consultation.") or `ARernstn` ("My concerns were taken seriously."). Note that this sample data contains only a small subset of the items addressed within the full PCI-questionnaire as it is only for illustration purposes.

The survey answers are coded through the patient selected category mapped on point values from 0 to 100. This is the desired $[0, 1]$-scale multiplied by 100 (re-scaling into $[0, 1]$ happens within the evaluation at a later step). Answers may also be missing, i.e. `NA`, if the patient did not select any answer. Answers may also take certain special values, e.g. the item `ARernstn` can take the value `-99` which means that the patient actively made 'no statement' on this item by selecting the corresponding option. As a general rule, negative values represent non-informative answers that cannot be used for further evaluation. To obtain an overview on a specific item, its survey question and answer possibilities, one can refer to the data documentation (`?iqtigprm::raw_data_pci`) for an English explanation or just check the attributes of the corresponding data column (descriptions in German).

[1] To enable a better understanding the items were translated for this tutorial. However, there is no validated translation of the questionnaire.

```
iqtigprm::raw_data_pci %>% pull(ARernstn) %>% attributes
#> $label
#> [1] "Ärztinnen und Ärzte: Mit meinen Anliegen wurde ich ernst genommen."
#>
#> $format.spss
#> [1] "F8.2"
#>
#> $class
#> [1] "haven_labelled" "vctrs_vctr"     "double"
#>
#> $labels
#>    Keine Angabe Weiß nicht mehr             Nie          Selten        Meistens
#>             -99             -98               0              33              67
#>           Immer
#>             100
```

Value mappings

The first step in data processing includes some cleaning of non-informative values and, if necessary, some remapping of the informative point values. This step requires a prespecified mapping list, which is also contained in the package.

```
iqtigprm::mappings_pci
#> $point_mappings
#> $point_mappings[[1]]
#> $point_mappings[[1]]$mapping
#>  25  50  75 100
#>  25  50  75 100
#>
#> $point_mappings[[1]]$felder
#> [1] "PAvorbeeintrn"
#>
#>
#> $point_mappings[[2]]
#> $point_mappings[[2]]$mapping
#>   0  33  67 100
#>   0  33  67 100
#>
#> $point_mappings[[2]]$felder
#> [1] "ARermutigtn"    "ARernstn"       "ARrespektn"     "ARgelegenheitn"
#> [5] "ARInfverstn"    "ARfachwortn"    "ARdeutschn"     "ARgesprochenn"
#>
#>
#> $point_mappings[[3]]
#> $point_mappings[[3]]$mapping
#>   0 100
#>   0 100
#>
#> $point_mappings[[3]]$felder
#> [1] "PAvoranginan" "Anginaruhen"
#>
#>
#> $point_mappings[[4]]
#> $point_mappings[[4]]$mapping
```

```
#>   0  50 100
#>   0  50 100
#>
#> $point_mappings[[4]]$felder
#> [1] "Anginaleichtn"  "Anginaschwern"  "Anginaaussergn"
#>
#>
#>
#> $ausweichkategorien
#> $ausweichkategorien[[1]]
#> $ausweichkategorien[[1]]$value
#> [1] -99
#>
#> $ausweichkategorien[[1]]$felder
#>  [1] "ARermutigtn"    "ARernstn"       "ARrespektn"     "ARgelegenheitn"
#>  [5] "ARInfverstn"    "ARfachwortn"    "ARdeutschn"     "ARgesprochenn"
#>  [9] "PAvoranginan"   "Anginaruhen"    "Anginaleichtn"  "PAvorbeeintrn"
#> [13] "Anginaschwern"  "Anginaaussergn"
#>
#>
#> $ausweichkategorien[[2]]
#> $ausweichkategorien[[2]]$value
#> [1] -98
#>
#> $ausweichkategorien[[2]]$felder
#>  [1] "ARermutigtn"    "ARernstn"       "ARrespektn"     "ARgelegenheitn"
#>  [5] "ARInfverstn"    "ARfachwortn"    "ARdeutschn"     "ARgesprochenn"
#>  [9] "PAvoranginan"   "PAvorbeeintrn"
#>
#>
#> $ausweichkategorien[[3]]
#> $ausweichkategorien[[3]]$value
#> [1] -97
#>
#> $ausweichkategorien[[3]]$felder
#> [1] "Anginaruhen"    "Anginaleichtn"  "PAvorbeeintrn"  "Anginaschwern"
#> [5] "Anginaaussergn"
#>
#>
#> $ausweichkategorien[[4]]
#> $ausweichkategorien[[4]]$value
#> [1] -96
#>
#> $ausweichkategorien[[4]]$felder
#> [1] "Anginaruhen"    "Anginaleichtn"  "PAvorbeeintrn"  "Anginaschwern"
#> [5] "Anginaaussergn"
#>
#>
#> $ausweichkategorien[[5]]
#> $ausweichkategorien[[5]]$value
#> [1] -90
#>
#> $ausweichkategorien[[5]]$felder
```

```
#> [1] "ARgelegenheitn" "Anginaleichtn"  "Anginaschwern"  "Anginaaussergn"
```

This list contains two objects `point_mappings` and `ausweichkategorien`. The object `point_mappings` is itself a list of different mapping rules and which fields they apply to. For instance, the first list object maps the values `25, 50, 75, 100` to themselves, respectively, and applies only to the column `PAvorbeeintrn`. This seems to be needless, but illustrates the possibility to map raw survey data to the desired points on the scale from 0 to 100, e.g. if the original answers in the raw data would be coded as answer values `1` to `4`.

The second sub-list `ausweichkategorien` defines the non-informative values for each item, which are mapped to `NA`. This is given by a set of lists containing a specific non-informative `value` and the fields (in German: `felder`) for which this definition applies. The mapping is applied by using the function `preprocess_data`.

```
processed_data_pci <- preprocess_data(raw_data = raw_data_pci, mappings = mappings_pci)
processed_data_pci %>% dplyr::glimpse()
#> Rows: 109
#> Columns: 22
#> $ ID             <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ~
#> $ ID_LE          <chr> "LE_A", "LE_A", "LE_A", "LE_A", "LE_A", "LE_A", "LE_A",~
#> $ Fragebogen     <dbl+lbl> 1, 2, 2, 1, 3, 2, 1, 2, 3, 1, 1, 2, 1, 1, 1, 3, 3, ~
#> $ FB_Sent        <date> 2018-02-12, 2018-02-13, 2017-12-31, 2018-02-04, 2018-0~
#> $ Gebdatum       <date> 1945-01-07, 1953-05-29, 1942-09-20, 1938-11-06, 1962-0~
#> $ Geschlecht     <dbl+lbl> 2, 2, 2, 2, 2, 2, 1, 2, 1, 1, 2, 2, 1, 1, 2, 1, 2, ~
#> $ BMI            <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
#> $ ARermutigtn    <dbl+lbl> 100, 100,  67,  67, 100,  67,  67, 100, 100,  NA, 1~
#> $ ARernstn       <dbl+lbl> 100, 100, 100, 100, 100, 100, 100, 100, 100,  NA, 1~
#> $ ARrespektn     <dbl+lbl> 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 1~
#> $ ARgelegenheitn <dbl+lbl>  NA, 100, 100, 100,  NA, 100,  67, 100,  NA,  NA, 1~
#> $ ARInfverstn    <dbl+lbl> 100,  33,  67, 100, 100,  67,  67,  67,  NA, 100, 1~
#> $ ARfachwortn    <dbl+lbl> 100,  33,  33, 100, 100, 100,  33, 100,  NA,  NA, 1~
#> $ ARdeutschn     <dbl+lbl> 100, 100, 100, 100, 100, 100, 100, 100,  NA, 100, 1~
#> $ ARgesprochenn  <dbl+lbl> 100, 100,  33, 100, 100, 100,  67, 100,  NA,  NA, 1~
#> $ PAvoranginan   <dbl+lbl>   0, 100,  NA,   0,  NA,  NA, 100, 100,  NA,   0,  ~
#> $ Anginaruhen    <dbl+lbl>  NA, 100,  NA,  NA,  NA,  NA, 100,   0,  NA,  NA,  ~
#> $ Anginaleichtn  <dbl+lbl>  NA,   0,  NA,  NA,  NA,  NA,   0, 100,  NA,  NA,  ~
#> $ PAvorbeeintrn  <dbl+lbl>  NA,  75,  NA,  NA,  NA,  NA,  50,  50,  NA,  NA,  ~
#> $ Anginaschwern  <dbl+lbl>  NA, 100,  NA,  NA,  NA,  NA, 100, 100,  NA,  NA,  ~
#> $ Anginaaussergn <dbl+lbl>  NA, 100,  NA,  NA,  NA,  NA,  50, 100,  NA,  NA,  ~
#> $ meta_unit      <chr> "LE_A", "LE_A", "LE_A", "LE_A", "LE_A", "LE_A", "LE_A",~
```

In the output `processed_data_pci`, the point values between 0 and 100 are left unchanged compared to the raw data whereas all former negative values are set to `NA`.

QI computation

In order to compute QI results, it requires QI computation specifications on how to evaluate the data. This comes in the form of 1) a quality indicator data base (QIDB) containing such computation rules for each QI and 2) a list of required precomputations, if necessary. In the following it is shown how these objects are incorporated into the whole evaluation work flow.

Precomputated fields

We start be showing how to apply specific precomputations to a data set. This is different from the previous preprocessing steps, as it takes the survey data set and adds newly computed columns, which can be later used up within the QI computation rules. Thus, these precomputations represent informative extensions to the data instead of only being a formal processing step. The package contains a list of precomputations for

the PCI example.

```
iqtigprm::precomputations_pci
#> $fn_QI56100_Index_NA
#> $fn_QI56100_Index_NA$expr
#> (PAvoranginan %==% 100 & ((is.na(Anginaruhen) & is.na(Anginaschwern) &
#>     is.na(Anginaleichtn) & is.na(Anginaaussergn)) | is.na(PAvorbeeintrn))) |
#>     is.na(PAvoranginan) | Fragebogen != 2
#>
#> $fn_QI56100_Index_NA$prototype
#> NULL
#>
#> $fn_QI56100_Index_NA$labels
#> NULL
#>
#>
#> $fn_QI56100_Index_value
#> $fn_QI56100_Index_value$expr
#> ifelse(fn_QI56100_Index_NA == TRUE, NA_integer_, ifelse(PAvoranginan %==%
#>     100 & (((Anginaruhen %==% 100 | Anginaleichtn %==% 100 |
#>     Anginaschwern %==% 100) & PAvorbeeintrn %>=% 50) | (Anginaleichtn %==%
#>     50 | Anginaschwern %==% 50 | Anginaaussergn %==% 50) | PAvorbeeintrn %>=%
#>     75), 100L, 0L))
#>
#> $fn_QI56100_Index_value$prototype
#> [1] "PAvoranginan"
#>
#> $fn_QI56100_Index_value$labels
#> NULL
```

The object `precomputations_pci` is a named list of several precomputations applied to the data set. Each component itself is also a list, where the name of each list gives the name of the corresponding column added to the data. The component `.$expr` provides the computation rule used to fill the new column. The components `.$prototype` and `.$labels` are additional arguments in order to define the labels of the newly computed column values.

This precomputation list is applied to the data set through the function `apply_precomputations`.

```
precomputed_data <- apply_precomputations(data = processed_data_pci,
                                          precomputations_list = precomputations_pci)
precomputed_data %>% dplyr::glimpse()
#> Rows: 109
#> Columns: 24
#> $ ID                    <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, ~
#> $ ID_LE                 <chr> "LE_A", "LE_A", "LE_A", "LE_A", "LE_A", "LE_A",~
#> $ Fragebogen            <dbl+lbl> 1, 2, 2, 1, 3, 2, 1, 2, 3, 1, 1, 2, 1, 1, 1~
#> $ FB_Sent               <date> 2018-02-12, 2018-02-13, 2017-12-31, 2018-02-04~
#> $ Gebdatum              <date> 1945-01-07, 1953-05-29, 1942-09-20, 1938-11-06~
#> $ Geschlecht            <dbl+lbl> 2, 2, 2, 2, 2, 2, 1, 2, 1, 1, 2, 2, 1, 1, 2~
#> $ BMI                   <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
#> $ ARermutigtn           <dbl+lbl> 100, 100,  67,  67, 100,  67,  67, 100, 100~
#> $ ARernstn              <dbl+lbl> 100, 100, 100, 100, 100, 100, 100, 100, 100~
#> $ ARrespektn            <dbl+lbl> 100, 100, 100, 100, 100, 100, 100, 100, 100~
#> $ ARgelegenheitn        <dbl+lbl>  NA, 100, 100, 100,  NA, 100,  67, 100,  NA~
#> $ ARInfverstn           <dbl+lbl> 100,  33,  67, 100, 100,  67,  67,  67,  NA~
#> $ ARfachwortn           <dbl+lbl> 100,  33,  33, 100, 100, 100,  33, 100,  NA~
```

```
#> $ ARdeutschn              <dbl+lbl> 100, 100, 100, 100, 100, 100, 100, 100,  NA~
#> $ ARgesprochenn           <dbl+lbl> 100, 100,  33, 100, 100, 100,  67, 100,  NA~
#> $ PAvoranginan            <dbl+lbl>   0, 100,  NA,   0,  NA,  NA, 100, 100,  NA~
#> $ Anginaruhen             <dbl+lbl>  NA, 100,  NA,  NA,  NA,  NA, 100,   0,  NA~
#> $ Anginaleichtn           <dbl+lbl>  NA,   0,  NA,  NA,  NA,  NA,   0, 100,  NA~
#> $ PAvorbeeintrn           <dbl+lbl>  NA,  75,  NA,  NA,  NA,  NA,  50,  50,  NA~
#> $ Anginaschwern           <dbl+lbl>  NA, 100,  NA,  NA,  NA,  NA, 100, 100,  NA~
#> $ Anginaaussergn          <dbl+lbl>  NA, 100,  NA,  NA,  NA,  NA,  50, 100,  NA~
#> $ meta_unit               <chr> "LE_A", "LE_A", "LE_A", "LE_A", "LE_A", "LE_A",~
#> $ fn_QI56100_Index_NA     <lgl> TRUE, FALSE, TRUE, TRUE, TRUE, TRUE, TRUE, FALS~
#> $ fn_QI56100_Index_value <dbl+lbl>  NA, 100,  NA,  NA,  NA,  NA,  NA, 100,  NA~
```

The output `precomputed_data` contains the two new columns fn_QI56100_Index_NA, fn_QI56100_Index_value, with values being derived from the computation rules. In particular, the field `fn_QI56100_Index_value` will be used in one of the provided example QIs.

Setting up Quality Indicators

The package contains a small QIDB `qidb_pci` containing two QIs as they were published[2] in IQTIG (2018). This includes the QIs:

- 56100: "Symptomatische Indikation aus Patientensicht bei elektiver PCI" (*"Symptomatic Indication from a Patient Perspective for elective PCI"*)
- 56105: "Prozessbegleitende Interaktion und Kommunikation der Ärztinnen und Ärzte" (*"Interaction and Communication with Physicians"*)

The data object `qidb_pci` is a list, where each component represents one QI, respectively. Each QI is a list itself containing the information necessary for computation.

```
iqtigprm::qidb_pci %>% .[[1]]
#> $Name
#> [1] "Symptomatische Indikation aus Patientensicht bei elektiver PCI"
#>
#> $KN_ID
#> [1] "56100"
#>
#> $GG
#> [1] "!fn_QI56100_Index_NA"
#>
#> $RefArt
#> [1] "Fest"
#>
#> $RefVal
#> [1] 0.95
#>
#> $RefOp
#> [1] ">="
#>
#> $Merkmale
#> $Merkmale[[1]]
#> $Merkmale[[1]]$Name
#> [1] "Symptomatische Indikation aus Patientensicht bei elektiver PCI"
```

[2]The QI "56100: Symptomatische Indikation aus Patientensicht bei elektiver PCI" has no reference value according to IQTIG (2018). For illustration purposes we add a reference value of 0.95 to the QI 56100 included in the example QIDB provided within this package.

```
#>
#> $Merkmale[[1]]$Items
#> [1] "fn_QI56100_Index_value"
```

This information includes the name, an id (`.$KN_ID`), specifications regarding the reference value (`.$RefArt` and `.$RefVal` and `.$RefOp`) and specifications on how to evaluate the data. The first one refers to the QI population (`.$GG`) and provides a specification on which cases are included or excluded for evaluation, i.e. only cases that fulfill the provided filter condition are included. The second computation specification is given through `.$Merkmale`, a list of the distinct 'Merkmale' and its associated items to be computed for the QI. Here each QI Merkmal is evaluated according to the Bayesian approach explained in section The Function theta_bayes(). The QI displayed here consists of only one Merkmal, which consists of only the binary (precomputed) item `fn_QI56100_Index_value`.

As a necessary step for evaluation, it needs to be defined on which data the QIDB should be evaluated. This is done by the function `connect_qidb2data`, which produces output that serves as input for the eventual evaluation function. The function `connect_qidb2data` not only binds the data to each QI, but also performs some preparations. For instance, it reduces the attached data to the relevant columns and patients for each QI, and also extracts some item information from the data and counts Merkmale and associated items and item levels for the QI.

```
qidb_data_pair <- connect_qidb2data(qidb = qidb_pci, data = precomputed_data)
qidb_data_pair %>%.[[1]]
#> $Name
#> [1] "Symptomatische Indikation aus Patientensicht bei elektiver PCI"
#>
#> $KN_ID
#> [1] "56100"
#>
#> $GG
#> [1] "!fn_QI56100_Index_NA"
#>
#> $RefArt
#> [1] "Fest"
#>
#> $RefVal
#> [1] 0.95
#>
#> $RefOp
#> [1] ">="
#>
#> $Merkmale
#> $Merkmale[[1]]
#> $Merkmale[[1]]$Name
#> [1] "Symptomatische Indikation aus Patientensicht bei elektiver PCI"
#>
#> $Merkmale[[1]]$Items
#> [1] "fn_QI56100_Index_value"
#>
#> $Merkmale[[1]]$L
#> [1] 1
#>
#>
#>
#> $M
#> [1] 1
```

```
#>
#> $L
#> [1] 1
#>
#> $Lvec
#> [1] 1
#>
#> $colNames
#>   fn_QI56100_Index_value
#> "fn_QI56100_Index_value"
#>
#> $item_levels
#> fn_QI56100_Index_value
#>                      2
#>
#> $observed_item_levels
#>      fn_QI56100_Index_value
#> [1,]                     NA
#> [2,]                    100
#> [3,]                      0
#>
#> $item_fragen
#> fn_QI56100_Index_value
#>             "Funktion"
#>
#> $item_weight
#> [1] 1
#>
#> $Auff_Stat
#> function(res) {
#>     with(
#>       q,
#>       if (RefOp == ">=") {
#>         as.numeric(res$interval["upper"]) <= RefVal
#>       } else {
#>         as.numeric(res$interval["lower"]) >= RefVal
#>       }
#>     )
#>   }
#> <bytecode: 0x0000000018c7d260>
#> <environment: 0x0000000018c86d90>
#>
#> $data
#> # A tibble: 17 x 4
#>       ID meta_unit fn_QI56100_Index_value w_fn_QI56100_Index_value
#>    <int> <chr>                       <dbl>                     <dbl>
#>  1     2 LE_A                            1                         1
#>  2     8 LE_A                            1                         1
#>  3    12 LE_A                            1                         1
#>  4    18 LE_A                            1                         1
#>  5    25 LE_A                            1                         1
#>  6    30 LE_A                            1                         1
#>  7    33 LE_B                            0                         1
```

```
#>  8    42 LE_B                          1                          1
#>  9    46 LE_B                          0                          1
#> 10    49 LE_B                          0                          1
#> 11    55 LE_B                          0                          1
#> 12    56 LE_B                          0                          1
#> 13    63 LE_B                          1                          1
#> 14    81 LE_B                          1                          1
#> 15    84 LE_B                          0                          1
#> 16    86 LE_B                          0                          1
#> 17   101 LE_B                          0                          1
#>
#> attr(,"class")
#> [1] "list" "qi"
```

In particular the object `qidb_data_pair[[1]]$data` is significantly reduced compared to the original data set as it only includes the columns for patient id and meta_unit as well as the column regarding the only QI relevant field `fn_QI56100_Index_value` and only the rows of the 17 patients with non-missing values in the QI.

Evaluation of the quality indicators

The `qidb_data_pair` is now ready to use for evaluation of the QIs, which is performed by the function `evaluateQI`. This function takes one component of the QIDB and data pair, i.e. one specific pair of QI and attached data, and computes the results for this QI applied to the data. As optional argument the function takes a `meta_unit`, i.e. a specific LE for which QI results should be computed. If a specific `meta_unit` is provided the data will be restricted to patients from that unit. Otherwise all patients within the data will be used to compute overall results.

Further arguments to the function are the prior parameters `a` and `b` for the Merkmal-specific beta prior distribution which are passed to the function `theta_bayes` that is applied to each Merkmal within `evaluateQI`. The argument `conf_level` gives the confidence level, i.e. the probability mass of the two-sided credibility interval to compute. The argument `nMC` sets the number of Monte-Carlo-Samples to use for computation of credibility intervals in the case of a QI with more than one Merkmal. (In that case posterior samples for each Merkmal are randomly drawn to generate a sample for the mean of the Merkmal-specific parameters and to compute sampling-basing credibility intervals. However the function `evaluateQI` still produces the same output for each repeated evaluation as it uses a fixed seed set within the function.)

```
one_QI <- qidb_data_pair[[1]]
one_LE <- "LE_A"
LE_result <- evaluateQI(one_QI,
                        meta_unit = one_LE,
                        conf_level = 0.95,
                        nMC = 1e5, a = 0.5, b = 0.5)

LE_result
#> $QI_hat
#> [1] 0.9285714
#>
#> $J
#> [1] 6
#>
#> $interval
#>     lower     upper
#> 0.6696111 1.0000000
#>
```

```
#> $Auff_stat
#> [1] FALSE
```

The function output `LE_results` contains only the most relevant results, i.e. the posterior mean estimate `.$QI_hat` and its associated credibility interval `.$interval.`, the number `.$J` of patients within the QI and an information whether the LE result is "statistisch auffällig", which means that the credibility interval lies entirely out of the reference domain (given through the reference value and direction, i.e. reference operator, of the QI).

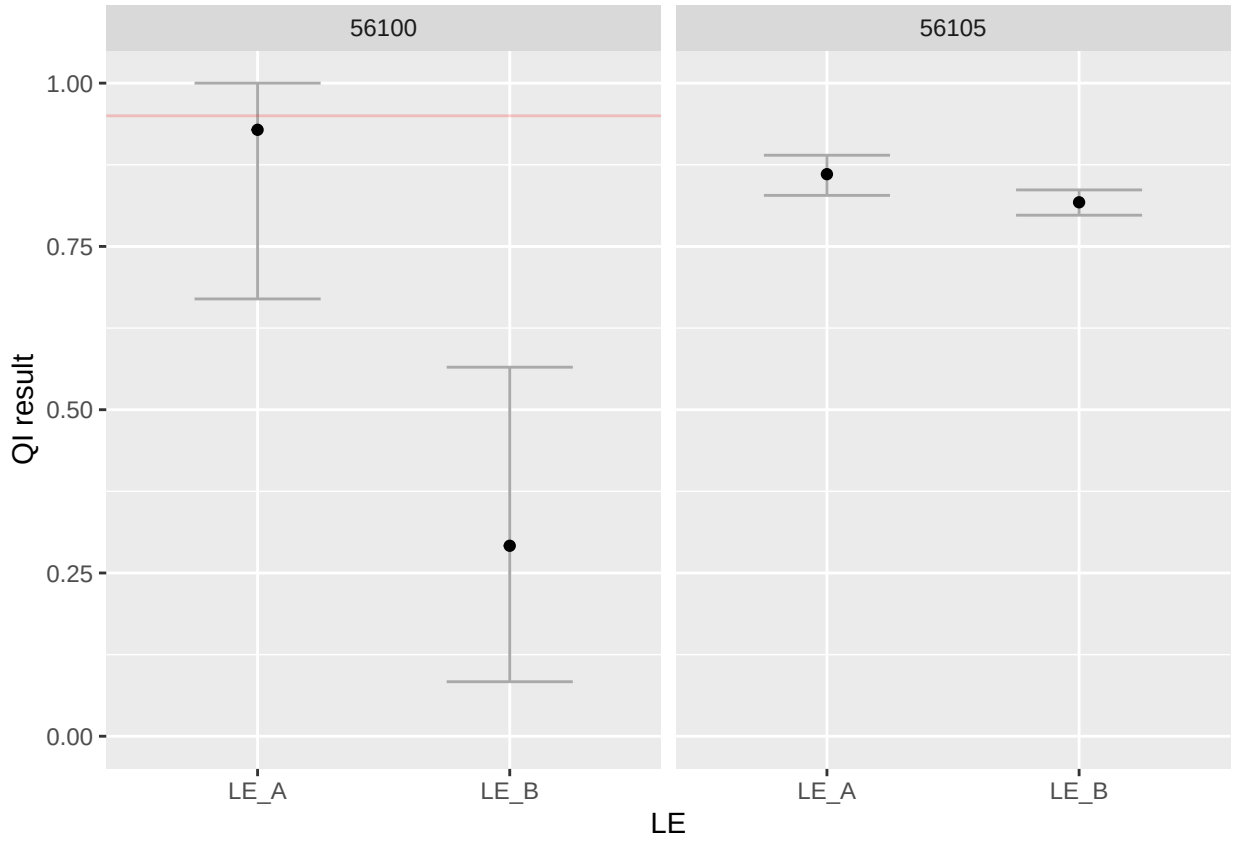
In order to compute the whole QIDB for the data, one can use `compute_results`, which just takes the `qidb_data_pair` as input and applies `evaluateQI` to all QIs in the QIDB and all meta_units in the data. The other arguments `conf_level`, `a`, `b` and `nMC` are as described above.

```
full_results <- compute_results(qidb_data_pair,
                                conf_level = 0.95,
                                a = 0.5, b = 0.5, nMC = 1e5)
#> QI 56100 for LE LE_A
#> QI 56100 for LE LE_B
#> QI 56105 for LE LE_A
#> QI 56105 for LE LE_B
full_results
#> # A tibble: 4 x 9
#>   KN_ID meta_unit QI_hat     J  lower upper Auff_stat RefOp RefVal
#>   <chr> <chr>      <dbl> <int>  <dbl> <dbl> <lgl>     <chr>  <dbl>
#> 1 56100 LE_A       0.929     6 0.670  1     FALSE     >=      0.95
#> 2 56100 LE_B       0.292    11 0.0835 0.565 TRUE      >=      0.95
#> 3 56105 LE_A       0.861    32 0.828  0.890 NA        >=     NA
#> 4 56105 LE_B       0.818    77 0.798  0.837 NA        >=     NA
```

This yields a data.frame output, where each row represents the output of one QI evaluation for one meta_unit, respectively. The QI and meta_unit are recorded within the first two columns `KN_ID` and `meta_unit`, followed by the associated results and finally some further information to the reference domain, given through `RefOP` and `RefVal`. In this example, the results for the QI `56105`, that does not have a reference value, do not obtain the `Auff_stat` classification.

The following figure shows these results, i.e. point estimates and uncertainty intervals for each LE on the $[0, 1]$-scale as well as a reference value if available (red line), stratified by QI given through the `KN_ID` as displayed above each plot.

```
full_results %>%
  ggplot2::ggplot() +
  ggplot2::geom_errorbar(ggplot2::aes(x = meta_unit, ymin = lower, ymax = upper),
                width = 0.5, color = "darkgrey") +
  ggplot2::geom_point(ggplot2::aes(x = meta_unit, y = QI_hat)) +
  ggplot2::geom_hline(ggplot2::aes(yintercept = RefVal), alpha = 0.2, color = "red") +
  ggplot2::facet_grid(. ~ KN_ID) +
  ggplot2::coord_cartesian(ylim = c(0,1)) +
  ggplot2::labs(x = "LE", y = "QI result")
#> Warning: Removed 2 rows containing missing values (geom_hline).
```



It becomes visually apparent, that that the result of `LE_B` for QI 56100 is "statistisch auffällig" as its credibility interval lies entirely below the reference value. In contrast `LE_A` is not classified as "statistisch auffällig" although its point estimate `QI_hat` is below the reference value.

References

IQTIG. 2018. "Entwicklung von Patientenbefragungen Im Rahmen Des Qualitätssicherungsverfahrens Perkutane Koronarintervention Und Koronarangiographie. Abschlussbericht. Stand: 15.12.2018." IQTIG [Institut für Qualitätssicherung und Transparenz im Gesundheitswesen]. https://iqtig.org/downloads/berichte/2018/IQTIG_Patientenbefragung_QS-PCI_Abschlussbericht_2018-12-15_barrierefrei.pdf.