

The R Package for Jackknife Confidentiality Protection

Intermediate Report for the ESSNet Project

Jobst Heitzig, December 2008

Goal

In this part of the project, we develop an open source software package named “condense” for the likewise open source R statistical computing environment (see <http://www.r-project.org>).

The package will be a reference implementation of basic methods of statistical analysis with included Jackknife confidentiality protection (see J. Heitzig, “The 'Jackknife' method: Confidentiality protection for complex statistical analyses”, Work session on statistical data confidentiality, UNECE, Geneva (2005), <http://www.unece.org/stats/documents/ece/ces/ge.46/2005/wp.39.e.pdf>). We plan to cover the most important uni- and bivariate analysis methods such as frequencies, sums, correlations, other moment- and quantile-based statistics, uni- and bivariate tests, etc., putting some focus on explorative methods.

This work is also related to the more general open source software development project “ConDense” (see J. Heitzig, “ConDense: towards an open source remote analysis system with jackknife confidentiality protection”, 56th Session of the International Statistical Institute, Lisboa (2007), <http://sourceforge.net/projects/condense>) which plans to use the developed R package in its back-end. In particular, we also use the ConDense code repository for versioning (see <http://condense.cvs.sourceforge.net/viewvc/condense/condense/R-packages/>)

Concept

Technical ingredients

Statistical computing environment

The computational statistics community largely uses the open source R statistical computing environment to develop, test, and share new statistical analysis methods and algorithms for its powerful, well-structured and transparent programming language and its interfaces to most common data storage systems (see, e.g., Rizzi, Alfredo and Vichi, Maurizio (Eds.): “COMPSTAT 2006 - Proceedings in Computational Statistics”, 17th Symposium Held in Rome, Italy (2006), <http://www.springer.com/statistics/computational/book/978-3-7908-1708-9>).

In this project, R was chosen as a development framework not only because of these reasons but also to make sure that the resulting software will be available freely to everyone and in the hope to stimulate some further cooperation with members of the computational statistics community.

Data storage

For performance reasons, many basic computations are however not performed by R itself but are delegated to a MySQL database instance, optimally running on the same hardware. Therefore also all data to be analyzed must reside in MySQL. Later versions will support

other data sources, too. MySQL was chosen for its free availability and wide-spread usage, in particular, it is usually installed and pre-configured on Linux systems. It is planned to let later versions of the package also store meta-data in MySQL as an alternative to specifying it in the function calls.

Communication with graphical or web front-ends

R functions can be called as web services via the standard protocol SOAP if their in- and output consists of sufficiently simple data types, using the R package “RSOAP” (<http://sourceforge.net/projects/rsoap/>). This is taken into account in the package design.

(For more information about the architecture of the larger “ConDense” project see http://condense.sourceforge.net/stcpm02_heitzig.pdf)

Design

Modularity and syntax

As R is a functional language, the package will consist of a number of top-level functions each performing a specific kind of statistical analysis, and of a number of bottom-level functions containing basic functionality common to these analyses and for performing the various steps of Jackknife confidentiality protection. While the former set of functions builds the interface for both the users of interactive R sessions and for graphical front-ends such as the ConDense front-end, the latter set of functions is designed for maximal extensibility to allow contributors of new statistical analysis methods to develop “safe” versions of their methods by combining them with the packages helper functions.

However, functionality and calling syntax of the top-level functions were also designed in a certain analogy to common procedures of the SAS® language, since it was felt that the SAS® procedures MEANS, UNIVARIATE, FREQ, and CORR more or less cover those methods most important for end-users, and since for some of these already a prototypical implementation of the Jackknife method had been developed in form of SAS® macros.

The basic calling syntax of the top-level functions is exemplified by the following call to the function “jk.means”:

```
jk.means (  
  input.table = “person”,  
  analyse = c(“age”, “income”),  
  group.by = c(“gender”),  
  where = “income>0”,  
  statistics = c(“mean”, “median”),  
  mu0 = 0.0,  
  [other optional parameters,]  
  result.format = “compact”  
  meta = list (  
    person = list(type=“mysql”, db=“census”, ...),  
    age = list(type=“integer”, confidential=false, ...),  
    income = list(type=“euros”, confidential=true, ...),  
    gender = list(type=“mf”, ...),  
    mf = list(scale=“nominal”, values=c(“F”, “M”), ...),  
    [more meta-data specifications]  
  )  
)
```

All function arguments are given by keyword parameters using numeric, alphanumeric, vector, and list data types only to make them easily callable through SOAP.

The parameters “input.table”, “analyse”, “group.by”, and “where” in the above example are common to many top-level functions and refer to meta-data items contained in the “meta” parameter (or, in a later version, alternatively contained in some meta-data-base in MySQL). These items describe the relevant attributes of objects like tables, columns, grouping hierarchies, according to the ConDense meta-data model: database connection data, scale and confidentiality of columns, levels and node labels of grouping hierarchies, and so on.

Other parameters, such as “statistics” and “mu0”, are specific to one top-level function.

Result formats

The return value of a top-level function is usually a multi-dimensional result table, returned as either a list containing several (nested) lists or vectors of headers and either a nested list of vectors or a vector-stream of data cells, both these formats using simple SOAP-friendly data types only. In a later version, data cells may also contain graphical objects (e.g. diagrams) encoded in svg, also.

Alternatively, the result may also be returned as a linearized table containing a column for each grouping level, which is more convenient in interactive R sessions or when the function is called inside another function which post-processes the results.

In the above example, these three result formats would look like this:

“Nested” format:

```
list (  
  dims = c(“Gender”, “Column”, “Statistics”),  
  labels = list( c(“F”, “M”), c(“Age”, “Income”), c(“Mean”, “Median”) ),  
  data = list(  
    list(  
      c(40.3, 41.5), c(36.4, 53.5)  
    ),  
    list(  
      c(42.5, 35.6), c(23.6, 52.5)  
    )  
  ),  
  slice.dims = c(“Statistics”),  
  slice.groups = c(“N”),  
  horizontal.dims = c(“Column”),  
  vertical.dims = c(“Gender”)  
)
```

“Compact” format:

```
list (  
  dims = c(“Gender”, “Column”, “Statistics”),  
  labels = list ( c(“F”, “M”), c(“Age”, “Income”), c(“Mean”, “Median”) ),  
  data.stream = c(40.3, 41.5, 36.4, 53.5, 42.5, 35.6, 23.6, 52.5),  
  slice.dims = c(“Statistics”),  
  slice.groups = c(“N”),  
  horizontal.dims = c(“Column”),  
  vertical.dims = c(“Gender”)
```

)

“Linearized” format:

Gender	Age_Mean	Age_Median	Income_Mean	Income_Median
F	40.3	41.5	36.4	53.5
M	42.5	35.6	23.6	52.5

Top-level functions

jk.means

Provides moment-based univariate statistics: number of observations, sum, uncorrected and corrected sum of squares, uncorrected sum of cubes or fourth powers, arithmetic mean and standard deviation with standard error and confidence limits, variance, t test statistics, skewness, kurtosis, bimodality coefficient, coefficient of variation, Blest's alpha4b-kurtosis and L4 statistics.

jk.univariate

Provides additional order-statistics-based univariate statistics: minimum, maximum, median, quartiles, other quantiles, range, quartile range, and robust measures of location (trimmed means, Tukey's biweight and trimean), scale (notches, median absolute deviation from the median), skewness and kurtosis. Will also draw box plots in a later version.

jk.freq

Provides statistics for two-way frequency tables, with (asymptotic) significance levels and confidence limits: chi-square (simple, likelihood-ratio, Mantel-Haenszel, continuity-adjusted), correlation coefficients (Pearson, Spearman, Kendall), contingency and uncertainty coefficients, gamma, lambda, Stuart's tau-c, Somer's D. For certain tables, also kappa, Bowker's test of symmetry, the Cochran-Armitage test for trend, phi, odds ratio, and Cramer's V.

jk.corr

Provides bivariate correlation coefficients (Pearson, Spearman, Kendall) and covariance. Will also draw two-dimensional box plots in a later version.

Bottom-level functions

jk_.univariate.prepare

Prepares everything needed for a univariate analysis by calling the following four functions. A similar function will exist for the bivariate case.

jk_.prepare.data

Connects to and prepares the data source and working space.

jk_.parse.group.by

Parses the group.by parameter and prepares the grouping hierarchies specified therein. Constructs an SQL view of the data in which each observation occurs once in each group it belongs to.

jk_.prepare.replacements

Constructs SQL code snippets for each confidential column which produce two replacement values from a random distribution controlled by five control-quantiles (q0,q1,q2,q3,q4) per

column given in the “meta” parameter. The replacement values y are computed from a standard normally distributed x by means of one of the following transformations:

If both q_0 and q_4 are infinite, $y = a + bx + c\sqrt{1+x^2}$ with (a,b,c) determined from (q_1,q_2,q_3) .

If q_0 is finite and q_4 is infinite, $y = q_0 + \exp(a+bx)$ with (a,b) determined from (q_0,q_2,q_3) .

If q_0 is infinite and q_4 is finite, $y = q_4 - \exp(a+bx)$ with (a,b) determined from (q_1,q_2,q_4) .

If both q_0 and q_4 are finite, $y = a + b\cdot\text{atan}(c+dx)$ with (a,b,c,d) determined from (q_0,q_2,q_3,q_4) if q_2 is nearer to q_0 than to q_4 , and from (q_0,q_1,q_2,q_4) otherwise.

jk_.make.seed.base

Uses the above SQL code snippets to provide an SQL view containing the replacement values. Produces the needed normally distributed x by MD5-hashing the original value, splitting it up into two pseudo-uniformly distributed parts and transforming them using the Box-Muller formula. Then constructs for each group a seed value based on the true and replacement values, to be used in the construction of safe publishing intervals.

jk_.univariate.finish

Composes the result structure for univariate analysis functions from SQL for true results and result variants provided by the top-level function, using the group seeds provided by `jk_.make.seed.base`. A similar function will exist for the bivariate case.

Status

Theory

Besides the remaining design and implementation issues, also some theoretical work remains to be done on the question of how the random seeds used for the construction of publishing intervals should be chosen precisely in the presence of where-conditions or pre-transformation of columns.

If one wants to enable the use of pre-transformations $t(x)$, such as translations, logs, powers, or signs, the seeds must neither depend too less nor too much on these transformations, otherwise a malevolent user could use certain query strategies (“scanning” or “shifting”). In a “scanning” attack, the user would e.g. repeatedly request the mean of $t(x) = \text{sign}(x - a)$ for all kinds of values a , searching for those values of a where the result changes. In a “shifting” attack, the user would e.g. repeatedly request the mean of $t(x) = x + a$ for all kinds of values a , then averaging $(\text{result} - a)$ to get a more precise value for the true mean.

Design and implementation

As of December 2008, the core functionality of most bottom-level functions and one top-level functions has been implemented and their syntax specified, and the meta-data model is almost finished. In irregular intervals, the latest version of the package is uploaded to the above-mentioned code repository. In 2009, there will be several intensified phases of implementation, so that it seems likely that the December 2009 deadline for a productive release can be met.