Descriptive Analytics Project

PLS Implementation in R

André Oliveira (2016082)
Bruno Santos (2016166)
Pedro Pires (2016299)

June 2017

1. CONTENTS

1. Introduction

For this project, we have implemented all the baseline features of PLS algorithm in R. All the mark improvement features from the list were implemented, except the last.

- ✓ Handling formative blocks plus
- ✓ Including remaining two inner model approximation methods plus
- ✓ Assessment measures:
- ✓ R-Squared indexes
- ✓ Dillon-Goldstein's rho
- ✓ Cronbach's alpha
- ✓ Communality indexes
- ✓ Redundancy indexes
- ✓ Average Variance Extracted
- ✓ Goodness-of-fit index
- ✓ Bootstrapping and individual parameter pseudo t-tests for statistical significance
- ✗ PLS consistent estimates

This report has the purpose to explain how our implementation works, the main functions constructed and the outputs objects

2. PLS implementation

This PLS implementation was divide among several files/functions. The file *PLS.R* has the main function of the model.

In order to the run the model, the user should configure the **Input Files** (data and inner/outer models) and PLS parameters (weighting scheme, tolerance).

Input files

- Data
- Inner model structure
- Outer model structure

There are no restrictions for the data file. If it has any NA, it will be imputed with mean or correspondent column;

Inner/Outer Model

- It's expected that both inner model and outer model be in the same excel workbook.
- The inner model structure should be placed in a sheet with the name INNERMODEL, and the outermodel in a sheet with the name OUTERMODEL.
- The structure of the models must be according with following examples, and should have the same structure for reflective and formative models:

Inner Model sheet

FROM	TO
IMAGE	EXPECTATION
IMAGE	SATISFACTION
IMAGE	LOYALTY
EXPECTATION	VALUE
EXPECTATION	QUALITY
EXPECTATION	SATISFACTION
QUALITY	VALUE
QUALITY	SATISFACTION
VALUE	SATISFACTION
SATISFACTION	LOYALTY

Outer Model Sheet

MV	LV
IMAG1	IMAGE
IMAG2	IMAGE
IMAG3	IMAGE
IMAG4	IMAGE
IMAG5	IMAGE
EXPE1	EXPECTATION
EXPE2	EXPECTATION
EXPE3	EXPECTATION
QUAL1	QUALITY
QUAL2	QUALITY
QUAL3	QUALITY
QUAL4	QUALITY
QUAL5	QUALITY
QUAL6	QUALITY
QUAL7	QUALITY
QUAL8	QUALITY
QUAL9	QUALITY
VALU1	VALUE
VALU2	VALUE
SATI1	SATISFACTION
SATI2	SATISFACTION
SATI3	SATISFACTION
LOYA1	LOYALTY
LOYA2	LOYALTY

- Both of the model must have only two columns;
- Considering the OUTERMODEL sheet, the first column must have the manifest variables and in the second column the associated latent variable.
- The INNERMDOEL sheet represents the connections between latent variables. In the first column are the LV that are predecessor of the LV in the second column.
- The names for the LV must be equal on both sheets.

Executing the model

To be able to execute the model it is necessary to load all the input files and add all the files needed as source. One must assure that, input data, innermodel, outermodel are data.frames.

In the section “Functions Description” all the parameters for all functions are explained in more detail.

For simplicity of use the implementation, the file *run.R* was added, which has the call for the main functions, including the loading of the files, necessary libraries and bootstrapping of the model.

After that, the user should run the advance.analytics.pls function, for example:

```
model <-
advance.analytics.pls(data=bank,innermodel,outermodel,wsheme="Factor",tolerance=0.001,mode="A", full=TRUE)
```

Cross validation with bootstrapping method

To assess the average and standard deviation of N samples the function bootstrap.statistic function shall be executed, as example:

```
boot <- bootstrap.statistic(data=bank, n_samples = 20, innerm=inner.m,
outerm=outer.m, tolerance=0.001)
```

This functions returns the estimators and standard deviation for the outer.loadings and path.coeficients for the n_samples.

To get the t-values and p-values of our metrics it is possible to use the function bootstrap.tstat for example:

```
tval <- bootstrap.tstat(boot,model)
```

3. INDEX FUNCTIONS

The list of meaningful functions developed are the following:

- advance.analytics.pls
- alpha.metric
- get.communality
- get.Dillon.rho
- get.GoF
- get.redundancy.indexes
- get.R.square
- get.sd
- input.means
- create.conection.matrix
- normalize
- normalize.weights
- path.coef
- stop.criteria
- update.weigths
- create.w.matrix
- create.cov.matrix
- centroid.scheme
- Path.scheme
- create.y.matrix
- create.z.matrix
- coefficients.R
- bootstrap.matrix.sample
- bootstrap.statistic
- bootstrap.tstat

4. FUNCTIONS DESCRIPTION

In this section is described the parameters and the returns of the main functions: advance analytics.pls and bootstrap.statistic and bootstrap.tstat.

advance.analyics.pls

Function that is used to run PLS algorithm and create the inner and outer model.

advance.analytics.pls@parameters:

data :	data.frame with the data to be processed
innermodel :	data.frame with the innermodel, same format as described above
outermodel :	data.fram with outermodel, same format as described above
wscheme :	represents the weighting schema to be applied. Can be "Factor", “Centroid”or “Path”.
tolerance :	value for the tolerate error of the approximation. Used as stop criteria.
mode :	estimation model, "A" for reflective, ”B” for formative
max_iter :	maximum number of iterations allowed. Used as stop criteria.
full :	if TRUE all the model assessment metrics are computed. If FALSE, only the PLS is computed.

advance.analytics.pls@return:

The set of meaningfull returns of the functions are all of the baseline implementation and all the features except the metrics from bootstrap. object is the return object of the function.

object$path_coefficients :	Matrix of path coefficients
object$outer_loadings :	Matrix of outer loadings
object$cross_loadings :	Matrix of cross loadings
object$ total_effects :	Matrix of total effects
object$inner_weights :	Matrix of inner_weights
object$tolerance :	Value of estimation error before stop
object$iterations :	Number of iterations
object$outer_weights :	Matrix of outer weights
object$weighting_scheme :	Weighting scheme used for the estimation
object$r.square :	RSquare metrics
object$.Gof :	Goodness of Fit metrics
object$Cronbach.alpha :	Cronback-Alpha metrics
object$Dillon.Goldstein :	Dillon Goldstein metrics
object$Redundancy.indexes :	Redundancy Indexes metrics
object$Communality :	Communality Indexes
object$path :	Data.frame with coefficients, and correspondent link information

bootstrap.statistic

This used is to do the crossvalition of results with bootstrapping

bootstrap.statistics@parameters:

data :	data.frame with the data to be processed
n_samples:	Integer number meaning the number of samples to be validated estimated
innerm:	data.fram with outermodel, same format as described above
outerm:	data.fram with outermodel, same format as described above
tolerance :	value for the tolerate error of the approximation. Used as stop criteria.

mode :	estimation model, "A" for reflective, ”B” for formative

bootstrap.statistics@return:

object is a generic name for the return object of the function.

object$mean :	estimator of PLS model weights resulting from bootstrap samples
object$sd :	standard deviation of model weights estimator

bootstra.tstat

Generation of t-statistics for the samples of bootstraping.

bootstrap.tstat@parameters:

advance.analytics.pls:	Return object of advance.analytics.pls function
bootstrap.statistic:	Return object of bootstrap.statistics function

bootstrap.tstat@return :

object is a generic name for the return object of the function.

object$t:	t value for the standard deviation and mean of samples and n_samples, for every path coefficient
object$p:	p value for the t value and n_samples, for every path coefficient