Package reshape2

Converting Data from Wide to Long Format

Contents

- 1. Understanding the difference between Long Format and Wide Format
- 2. Introduction to reshape2
- 3. Reshaping data using cast and melt functions

Long format vs Wide format

Some data have wide format and some have long; for different analysis you may choose one of the either.

In Long Format data,

one column contains all the possible variables, another column contains their respective values. For example:

ID	Names	Subject	variable	value
1	Rohit	Maths	I_semester	78
2	John	Physics	I_semester	56
3	Vivek	Statistics	I_semester	76
4	Martina	Physics	I_semester	89
	_		I_semester	67
1	Rohit	Maths	<pre>II_semester</pre>	88
2	John	Physics	<pre>II_semester</pre>	59
3	Vivek	Statistics	<pre>II_semester</pre>	81
4	Martina	Physics	<pre>II_semester</pre>	73
5	Agatha	Statistics	<pre>II_semester</pre>	80

In Wide Format data,

each column represents a

different variable. For example:

ID	Names	Subject	I_semester	II_semester
1	Rohit	Maths	78	88
2	John	Physics	56	59
3	Vivek	Statistics	76	81
4	Martina	Physics	89	73
5	Agatha	Statistics	67	80

Long format vs Wide format

- Long data is mainly used in data visualization (plotting graphs) and statistical modeling. Also it is easy to perform operations on subsets of data.
- Wide data is used in the calculation of growth, for example, we have a company data with list of companies and their revenue for 4 Quarters. Our data is in wide format with 5 columns: Company, Quarter1, Quarter2, Quarter3, Quarter4. Since our data is in a wide format it is easy to calculate Quarter wise growth.
- In practical scenario, while wide format is more readable, long format is easier to analyze. Therefore, it is useful to know how to convert between the two.

Introduction to reshape2

- We know that data comes in many forms. Hence, we are required to reshape it according to our need.
- The process of reshaping data in R is tedious. R base functions for 'aggregation' reduces and rearranges the data into smaller forms, but with reduction in amount of information. Also, these functions don't solve the problem of converting data from wide to long format & vice versa. The package **reshape2** overcomes these problems.
- **reshape2** is an R package written and maintained by Hadley Wickham.
- reshape2 makes it easy to convert data from long to wide formats & vice versa.

Package reshape2 has two key functions:

melt – converts data from wide format to long format

cast – converts data from long format to wide format

Data Snapshot

stud_data consists student names & their marks scored in Math's, Economics & Statistics. It has 5 rows & 5 columns.

					/ariables				
		Studen	t ID	Names	Maths	Eco	nomics	Statistics	F
?	Г	1		Rohan	67		56		
		2		John	89		22	79	
Observations	Columns De		De	scription	cription Type		Measu	rement	Possible values
	Student_l D		St	udent ID	character				-
	Names Stud		lent names	chara	ter	-		-	
	Maths Mark		ks scored in Maths nume		ric	-		positive values	
			ks scored in conomics numeric		ric	-		positive values	
				s scored ir tatistics	nume	ric			positive values

Tasks to be Performed

stud_data is originally in a wide format. We will perform following activities using package reshape2:

- Convert data format from wide to long & Changing the names of new columns that will be created in the conversion process.
- Convert data format from long to wide. While doing this conversion perform subsetting and aggregation.

melt()

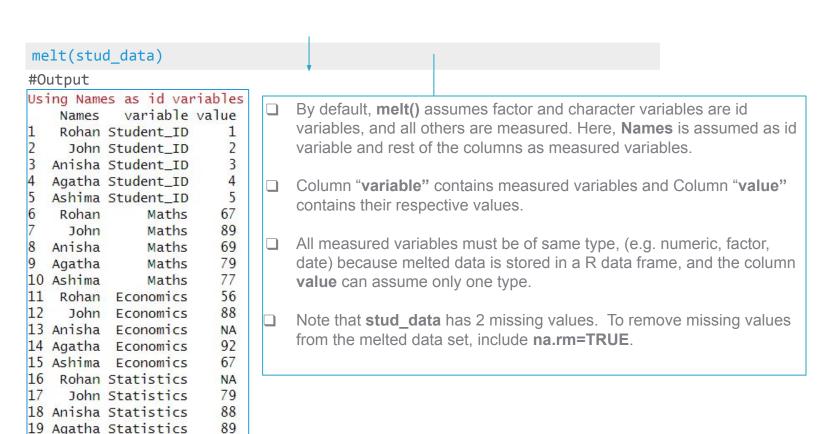
- melt() converts data from wide format to long format. It's a kind of restructuring where categorical variables are 'melted' into unique observations.
- Here variables are divided into 2 groups: identifier and measured variables.
- Identifier variables(id) identify the unit or column that measurements takes place on.

```
# Install and load package reshape2
# Import stud_data
# Convert wide data into long
install.packages("reshape2")
library(reshape2)
stud data<-read.csv("stud data.csv",header=TRUE)</pre>
```

melt()

20 Ashima Statistics

89



melt() with id.vars

ID variables can be separately specified

```
# Change the ID variables to 'Student ID' and 'Names'
melt(stud_data,id.vars=c('Student_ID','Names'))
#Output
                      variable value
   Student_ID
              Names
              Rohan
                         Maths
                                  67
               John
                                  89
                         Maths
                                       With id.vars= we can give
           3 Anisha
                    Maths
                                 69
                                  79
           4 Agatha Maths
                                       multiple id variables & we
           5 Ashima
                         Maths
                                       can specify them in a list of
                                  56
              Rohan Economics
                                  88
               John Economics
                                       variables enclosed in c().
           3 Anisha Economics
                                 NA
                                  92
           4 Agatha Economics
                                       If we only specify id
                                  67
10
           5 Ashima Economics
11
           1 Rohan Statistics
                                 NA
                                       variables, by default non id
12
                                  79
               John Statistics
13
                                  88
           3 Anisha Statistics
                                       variables will be taken as
14
                                  89
           4 Agatha Statistics
                                       measured variables.
15
           5 Ashima Statistics
                                  89
```

See the next example for setting measured variables.

melt() with measure.vars

Specify measured variables as Maths and Economics

	<pre>(stud_data, nomics'))</pre>	id.vars	=c('Studen	t_ID','	Names'),measure.vars=c('Maths',
#Outp	out Student_ID		variable		•
2 3 4 5	4	Rohan John Anisha Agatha Ashima	Maths	67 89 69 79 77 56	With measure.vars= one can specify a vector of measured variables.
7 8 9	4	John Anisha Agatha	Economics Economics Economics Economics	88 NA 92 67	

melt() with variable.name and value.name

What if we want to control the column names in our long format data? We can do that in just one step:

Change the names of new columns created after melting stud data long format<-melt(stud data,id.vars=c('Student ID','Names'),</pre> variable.name='Subjects', value.name='Marks') long format #Output Student_ID Subjects Marks Names Maths 67 Rohan John Maths 89 Using variable.name= 3 Anisha Maths 69 4 Agatha Maths and value.name= 5 Ashima Maths 56 Rohan Economics arguments, we can John Economics 3 Anisha Economics NA change the names of the 92 4 Agatha Economics 67 columns "variable" and 10 5 Ashima Economics 11 Rohan Statistics NA "value". 79 12 John Statistics 13 88 3 Anisha Statistics 14 4 Agatha Statistics 15 5 Ashima Statistics 89

cast function

cast function converts data from long format to wide format. It is a complement
of melt() function. There are two types of cast functions: dcast() and acast()

dcast() - returns a dataframe as the output

acast() - returns a vector/matrix/array as the output

The basic arguments of *cast is the data in long format and a formula of the form:

$$x1 + x2 \sim y1 + y2$$

id variables are specified on the left and measured variables on the right.

The order of the variables matter, the first varies slowest, and the last fastest.

Since dataframe objects are the most common, we will explore the dcast()
 function.

Let's take the **long-format** data object which we created in the previous example and cast it into some different wide formats.

dcast() with Formula

```
# Reshape long format object into wide format
dcast(long format,Student ID+Names~Subjects)
#Output
Using Marks as value column: use value.var to override.
  Student ID Names Maths Economics Statistics
              Rohan
                        67
                                  56
                                             NA
               John
                                             79
            3 Anisha
                                             88
                                  NA
                                  92
           4 Agatha
                                             89
           5 Ashima
                                  67
                                             89
```

dcast() uses a formula to describe the shape of the data.

```
# There are different ways in which you can dcast data.
```

See the below commands :

```
dcast(long_format, Names+Student_ID~Subjects)
dcast(long_format,Student_ID~Subjects)
dcast(long_format,Names ~ Subjects)
```

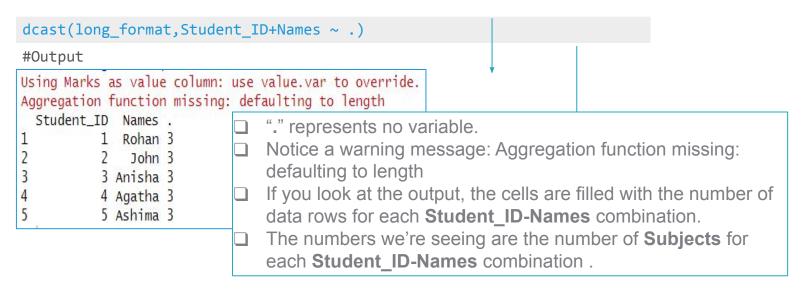
dcast() with subset

You can subset variables while converting data from long to wide format.

```
# Subsetting
# Reshape long format object into wide format displaying columns:
# Student ID, Names and Maths column
install.packages("plyr")
library(plyr)
dcast(long format,Student ID+Names~Subjects,
subset=.(Subjects=="Maths"))
#Output
Using Marks as value column: use value.var to override.
  Student ID Names Maths
         1 Rohan
                    89
             John
                                Here, we have sub setted Maths from the
          3 Anisha
                    69
                    79
                                variable Subjects.
          4 Agatha
          5 Ashima
                                   subset= is used to subset data using .()
                                   package plyr is needed to access '.'
```

dcast() with fun.aggregate

One confusing "mistake" you might make is casting a dataset in which there is more than one value per data cell. For example, this time we won't include any measured variable.



So, when you cast your data and there are multiple values per cell, you also need to tell **dcast** how to aggregate the data. See the next example for using aggregation function.

dcast() with fun.aggregate

Calculate total marks for all students

Aggregation

```
dcast(long format, Student ID+Names ~ .,fun.aggregate=sum,na.rm=TRUE)
Using Marks as value column: use value.var to override.
 Student ID Names
            Rohan 123
                       fun.aggregate is used to specify the
         2 John 256
         3 Anisha 157
                          aggregation function
         4 Agatha 260
                          na.rm=TRUE is to remove NA values.
         5 Ashima 233
                         In base R, aggregate() also has similar formula
                          where variables on left hand side are
                          continuous and on right hand side are
                          categorical.
```

In this case, an aggregation function reduces multiple values to a single one.

Quick Recap

In this session, we learnt how to change the data from long to wide format and vice-versa. Here is the quick recap:

Long vs Wide Format

- In Long Format data, one column contains all the possible variables, another column contains their respective values.
- In Wide Format data, each column represents a different variable.

melt()

- melt() converts data from wide format to long format.
- Usage: melt(data,id.vars,measure.vars, variable.name="variable",na.rm=FALSE, value.name="value")

cast()

- Use **acast** or **dcast** depending on whether you want vector/matrix/array output or data frame output.
- · Usage: dcast(data, formula, fun.aggregate=NULL, subset = NULL)