Qualitative variables

Data Crunchers

season 1 /episode 12

When you analyze data you will come across many types of variables. Very often you will deal with qualitative values and in some cases you will need to transform a quantitative variable into a qualitative variable.

In this episode you will learn:

* Which variables/ characteristics are referred to as qualitative characteristics?
* What basic operations can be performed on qualitative characteristics?
* How to summarize/ describe qualitative characteristics?

We will use two data sets to illustrate these issues. The first set, koty\_ptaki is quite small, while the other one, auta2012 is a much more extensive data set. Both data sets are available in the package PogromcyDanych.

What does it mean: a qualitative variable

Qualitative variables are categorical variables –they specify groups to which the data belongs. For that reason they are also called grouping variables. As this type of variable is referred to as factor in R, in Polish we use a loan translation zmienna faktor. Another name frequently used by programmers is zmienna czynnikowa (factor variable).

When we define a qualitative variable we should indicate what groups we are referring to. Examples of qualitative variables are e.g. education (groups: primary, secondary, higher education), gender (groups: male, female), country of residence (groups: Poland,...).

A qualitative variable can be expressed in numbers, yet unless these numbers stand for amount of size, such a variable is in reality a qualitative variable. For example, the first two numbers of a postal code may be a qualitative variable, while a school grade may be treated as a qualitative variable (or a quantitative variable depending on the context). The same concerns the number of car doors –if we want to use this characteristic to divide cars into two groups of two-door and three-door models, then this variable is treated a s a qualitative variable.

Although qualitative variables can greatly differ as far as number and type of groups are concerned, there are several operations which can be performed on all of them. Let us see what are these operations.

Loading data

Our first examples will be based on the data set koty\_ptaki from the package PogromcyDanych. In order to load this data you only need to activate the package –the instructions can be found in the episode 2.

Let us activate the package and use the function head() to display the first six rows. Which of the displayed variables are qualitative variables?

Each row describes one species. In this case all non-numerical variables are qualitative variables, that is, gatunek, habitat and druzyna (species, habitat and group).

Qualitative variables

Let us look at two variables –gatunek and druzyna.

When we display these variables, we can also see a set of all the potential values. This set is usually called levels of the qualitative variable or a dictionary of the variable.

There are two basic operations which we can perform on qualitative variables. We can read levels of the variable (with the function levels()) and we can determine the number of individual levels (with the function table()).

We can check whether a given variable is qualitative or quantitative using the function class(). If as a result we receive the word factor, we are dealing with a qualitative variable. If we get the word integer or numeric, we are dealing with a quantitative variable.

Qualitative variables

When we apply the two functions to two random variables from the data set koty\_ptaki we receive the following results.

Result of the function levels() is a vector of words –levels of the qualitative variable.

Result of the function table() is a vector of quantity of each factor.

Both variables differ in many aspects. There are only two possible values for druzyna and thirteen for gatunek (each row is a different value).

Loading data

The data set koty\_patki consists of 13 rows. All of them can be displayed on the computer screen. For such small sets we do not need special descriptive statistics to get the gist of their content.

For this reason from now on we will practice working with qualitative variables ? on a much bigger data set containing over 200 thousand values called auta\_2012, which is also available in the package PogromcyDanych.

You can find a description of this data set in the episode https://rawgit.com/pbiecek/MOOC/master/0\_dane/0\_dane.html

Let us load this set and look at its first two rows.

Which of the variables are qualitative variables?

Descriptive statistics

Now we will focus on characteristics such as Waluta or Marka. These characteristics are expressed by values from a finite list of possible values. Potential values of the characteristic Waluta are levels(auta2012$Waluta)= CHF, CZK, EEK, EUR, GBP, HUF, PLN, USD.

We can perform several operations on qualitative variables. One of them is displaying the list of possible values. That list is often called a dictionary (list of potential values).

You can display the list of possible values for a qualitative variable using the function levels().

Frequency distribution table is frequently used to describe qualitative variables. It displays number of occurrences of each value from the dictionary. You can draw the frequency distribution table with the function table() or summary(). If there are gaps in the qualitative variable, the function summary() will also specify their number.

Exercises

* How many variables from the data set auta2012 are qualitative variables?
* The variable liczba.drzwi assumes values 2/3 and 4/5. It is treated as a qualitative variable. Yet we can also assume that liczna.drzwi is a quantitative variable because it specifies number of doors in a car. How can you explain that duality?

You can find sample answers at https://rawgit.com/pbiecek/MOOC/master/0\_dane/9\_zadania.html

Percentages

In some cases it is much more convenient to use a table with percentage values rather than a frequency distribution table.

But how can we calculate these percentage values?

It can be done in several ways. The function prop.table() divides the values form the frequency distribution table by the sum of all the occurrences –it transforms quantities into fractions. Now let us attribute the result of the function table() to some variable, e.g. variable called waluty (we can choose any name but afterwards we have to use it consistently). Then we transform our newly created frequency distribution table into fractions.

In the following example we will put words? before and after /bracket the command attributing the result so that its result was displayed. If there were no brackets, the result of the function table() would not be displayed. Instead, it would be attributed to the variable waluty.

Percentages

The function prop.table() determines fractions which unfortunately are not really readable for humans. It would be easier for us to receive results as percentages. Not any percentages, but nicely formatted percentages e.g. displayed to one decimal place.

If you want to change fractions into percentages, you only need to multiply them by 100. In order to display the results to one decimal place you need to use the function round() which rounds the numbers down to specified decimal place.

All the operations presented above work for individual values (called scalars) as well as for vectors. For example, the function 100\*frakcje multiplies the number 100 by each element of the vector frakcje. The result of that operation is a factor. Not every programming language allows for such mixed operations multiplying vectors by scalars.

Ordered percentages

Now we have the percentage values. It is good to increase readability of this form of presentation by arranging the values in an ascending or descending order.

Even if we deal with only eight numbers, we may have problems with identifying the biggest number when these numbers have got many digits.

You may use the function sort() to order the numbers. When you also add the argument decreasing = TRUE, you will order the values in a descending order.

Ordered percentages

Values in the vector can be sorted in many other ways. The function order() is more complex but it will allows us to discuss more advanced examples of indexing.

The result of the function order() are indices of elements in a vector listed in an ascending order. You can find more information about this function in the episode ......????

If we read the vector zaokrogloneProcenty in the following order: first element, third, fifth, sixth, eighth, fourth, second and seventh, we will get an ordered sequence.

We can use the result of the function to index the original data set. In this way we can sort the whole vector.

But how can we check what are the positions of the three smallest values?

Exercises

* The characteristic Marka defines the make of the car. Check which make is the most popular one.
* The characteristic Rodzaj.paliwa defines the type of fuel used by the car. Is it petrol, diesel oil or gas? Check what percentage of cars runs on petrol.

You can find sample answers at https://rawgit.com/pbiecek/MOOC/master/0\_dane/9\_zadania.html

Graphic descriptive statistics

Frequency distribution tables are easy to interpret. Just like in other cases, however, graphic presentation of numbers allows for even easier comprehension of the content of data. All in all, it is fairly easy to make a mistake concerning number of digits in a number in a traditional table. On a diagram such values literally hit us in the eyes.

Frequency distribution tables are usually presented in the form of bar charts. They can be created e.g. with the function barplot().

Graphic descriptive statistics

If you want the bars to be oriented horizontally, you need to add the argument horiz = TRUE to the function barplot().

The values are listed in alphabetical order by default but this arrangement does not always make sense. In this case it is much more reasonable to sort the data in an descending order. You can do that using the function sort().

Use the argument las=1 to ensure that labels are oriented horizontally.

Exercises

* Create a graphic presentation of the frequency distribution table for the variable Kraj.pochodzenia. Try out the horizontal and vertical versions of the diagram. Add the argument las=1 in order to change the direction of the labels on the axes.

You can find sample answers at https://rawgit.com/pbiecek/MOOC/master/0\_dane/9\_zadania.html

Strings or factors?

Qualitative variables (factors) resemble string variables on the surface. Underneath, however, their representation is completely different. Factors are not saved as strings but as pairs number vector-dictionary, which specify which string corresponds to which number. In this way you can use less memory to save big data sets.

We also usually perform different operations on strings than on qualitative variables although it may happen that we deliberately try to read some data as strings. When we load data from a text file with strings, they are treated as factors by default.

In the example below variables such as gatunek and habitat are loaded as factor-type variables. The side effect is the manner in which the results are displayed by e.g. the variable cat(). This function will display on the screen numbers, not strings, for a qualitative variable.

Strings or factors?

The function read.table() has got an argument stringsAsFactors which specifies how the tables with characters should be displayed. When we specify the argument stringsAsFactors=FALSE we force the program to load data as string variables. They will occupy more space but the process of loading will take less time.

Notice the new version of the result of the functions str() and cat().

This time the column gatunek was loaded as a string vector.

Strings or factors?

Factor variables can be converted into characters with the function as.character().

Strings can be converted into factor variables with the function as.factor

Let us see how these functions work on the example.

Frequency distribution tables

What if we wanted to present the relationship between two qualitative characteristics? We could do that using the function table() (subsequent variables should be given as arguments), ????? or the function xtabs() which is bit more convenient to use.

The first argument of that function is the formula with the following syntax ~ zmienna\_1 + .. + zmienna\_n. It begins with a tilde and contains variables separated by the sign of plus. The second argument is a data frame which should include the variables listed in the formula.

The result is a contingency table with number of dimensions equal to the number of variables in the formula.

For example, if we juxtapose the country of origin and currency we get a frequency distribution table containing 32 rows and 8 columns. Lack of any names in the first row of the table corresponds to the NA value of the country of origin.

Frequency distribution tables

Frequency distribution tables are not data frames.

When they contain only two variables they indeed look like data frames –they have rows and columns. Yet they must be something different because when a contingency table concerns three or more variables, the resulting contingency table has more than two dimensions.

The function as.data.frame serves to transform multidimensional tables into data frames. It produces the so-called sparse contingency matrix in which first columns describe all the combinations of variable factors and the last column describes frequency distribution of individual combinations of factors.

Transformation of quantitative variables into qualitative variables

We can frequently facilitate understanding of quantitative data by dividing variables into ranges. Such ranges are themselves qualitative variables.

You can convert a qualitative variable into a quantitative variable using the function cut(). Its first argument is a quantitative variable and its second argument is number of ranges to be created or border points of these ranges.

In the example presented below we will transform the variable waga from the data set koty\_ptaki into four rages: 0-1, 1-10, 10-100 i 100-1000. The created variable will be then added to the data set koty\_ptaki under the name wagaKategoria.

Frequency distribution tables

We will use the newly created variable to present graphically the statistics for a pair of qualitative variables. We will juxtapose weight and group variables.

Then we will apply the function mosaicplot() to the two-dimensional table.

It draws a so-called mosaic plot which simultaneously presents multiple relationships. Firstly, it shows columns Kot and Ptak. Their width corresponds to the proportions of the values in the data set (their number is almost equal and for that reason the width of the columns is also almost equal). Secondly, height of rows corresponds to the relative share of particular values in a given column. In the Kot column the weight range 10-100 is domineering, while in the Ptak column it is the range 1-10, which is marked with blue.

This is one of the ways in which we can present conditional distributions. It is very useful although it requires some training in reading. You will be able to find more information about mosaic plots and other methods of presenting two qualitative variables in the episode 3.

Order and names of factors

The lexicographic order is the default order of the factors. When we create diagrams we usually present values in the order described by the attribute levels.

Order and names of factors

The default order can be altered. One of the ways in which we can specify the order of factors is to place the argument levels in the function factor(). The factors will be listed in the order indicated in that argument.

Order and names of factors

If you wish to sort the levels with respect to certain quantitative value, you can use a very convenient function reorder(). Its arguments are a vector with a factor variable, a vector with any variable and a function. For each group defined by the factor variable and for values determined by the second argument the value of the function -the second argument –is specified. Then the factors are sorted in accordance with the order of the results of the third function.

In the following example types of fuel are ordered by the average year of manufacture of cars running on a certain type of fuel. As you could expect, more ‘modern’ fuels are used in newer cars.

Order and names of factors

We can change not only order of factors but also their names. The most easiest way to do that is to use the function levels() as illustrated by the following example.

Exercises

* Divide the variable Rok.produkcji from the data set auta2012 into the following ranges: 1900-1990, 1991-1995, 1996-2000, 2001-2005, 2006-2010, 2011-2012 and the variable Przebieg.w.km into ranges: 0-1000, 1001-10 000, 10 001-100 000, 100 001 - 1000 000, 1000 000 - 10 000 000.
* Create a frequency distribution table for the two newly created variables.
* Present the table graphically.

You can see sample answers at: https://rawgit.com/pbiecek/MOOC/master/0\_dane/9\_zadania.html