**Logic Features**

pogRomcy danych (*Data Masters*) Season 1 / Episode 13

All rights reserved.  
The use shall be subject to prior consent of the author.

Press A to see the plain text instead of the slides.  
Press T to display the table of contents.

**What Is This Episode About?**

When analysing the data, we encounter different types of variables. We discussed the quantitative and qualitative ones. A specific type of the qualitative properties is the logic features in the form of the true/false logic values.

In this Episode, we will learn:

* What variables/features are called the logic features.
* What basic operations can be performed on the logic features.
* How to sum up/describe the logic features.

To illustrate these issues, we will use two sets of data. The first small set of data is koty\_ptaki and the second, considerably bigger one is auta2012, both available in the PogromcyDanych package.

**What Does Logic Feature Mean?**

The logic features are the features that describe one of two logic values: true or false. For example: is he taller than 150 cm, is it raining, did the patient live five years after a surgical procedure, etc.

The logic variables often result from processing the quantitative or quantitative variables. If three levels of education are described, an artificial feature ‘does he holds a university degree’ can be constructed. And with the height described as a quantitative feature, a logic feature ‘is he taller than 150 cm’ can be constructed.

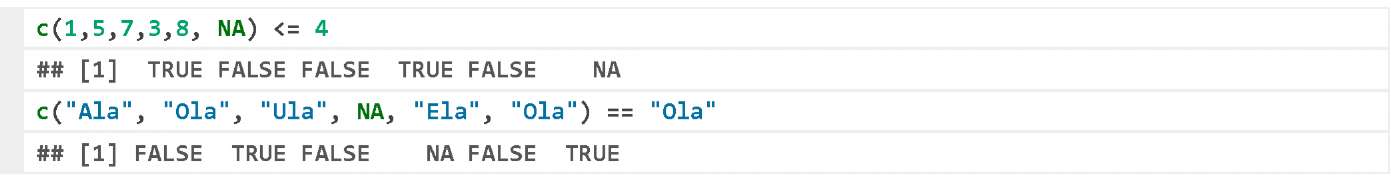
The logic variables can be considered the qualitative variables, since they take the values of the two-element dictionary. Both for the qualitative and logic values, a frequency table can be determined.

Several additional logic operations are available for the logic variables, which cannot be performed on the typical qualitative variables.

**Three-Layer Logic**

We have written before that the logic values take the following values: TRUE (logic truth) or FALSE (logic falsehood). At the same time, however, each type of the variables in the R program can also take an indeterminate value NA. In fact, the logic values operate in the three-layer logical structure.

In creating the logic values, we often use the logic operators > (greater), >= (greater equal), == (equal), <= (less equal), < (less). If you compare an indeterminate value NA with such an operator, the result will also be an indeterminate value.



In creating the logic value vectors instead of the full names TRUE and FALSE, you may also use the abbreviations T and F.



The special logic value operators include | (logic ‘or’) and & (logic ‘and’). The former returns the value of TRUE, if any of the arguments is true. The latter returns the value of TRUE, if both arguments are true.

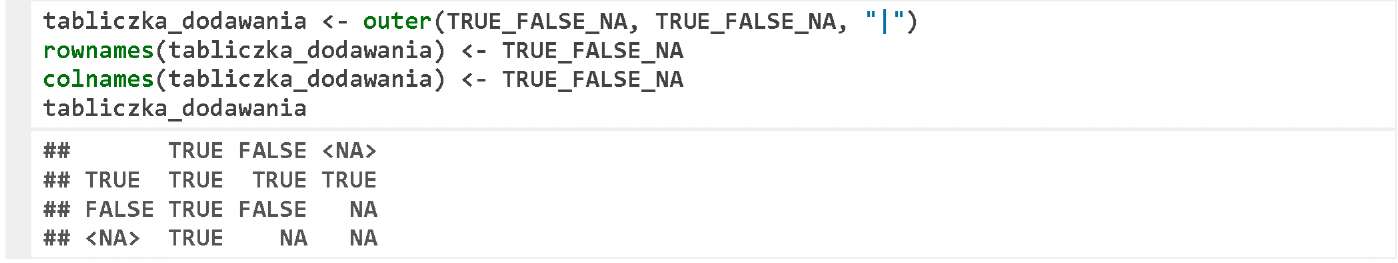


**Table of Logic Addition and Logic Multiplication**

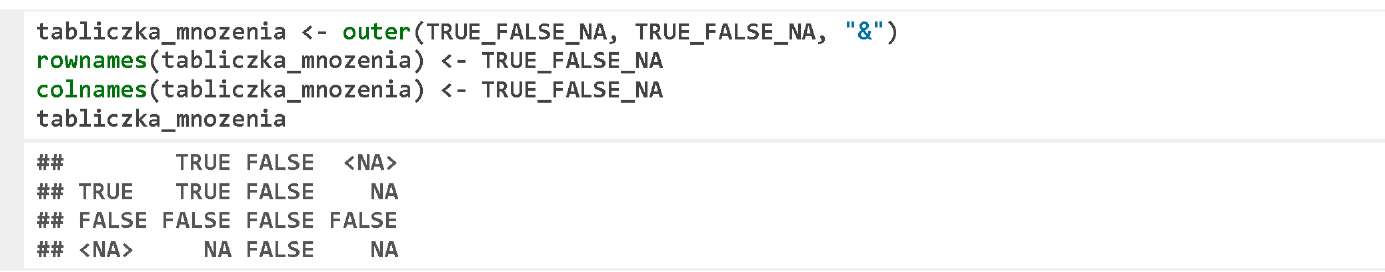
A useful tool to present the possible results of the | operator is a logic addition table. To present it, we will define a vector containing any of three possible logic values.



The function outer() adopts two vectors as the arguments, and next applies a function, which is the third argument, to each pair of the first and second vector values. Two following lines contain the names of the rows and columns to generate a neatly looking logic addition table. If any of the arguments features logic truth value, then the result of or is a logic truth.



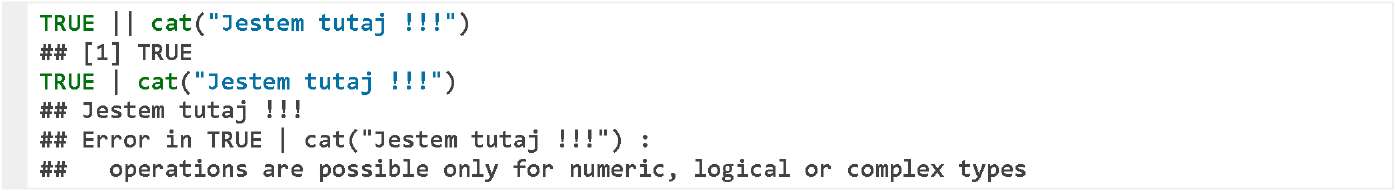
Similarly, we can construct a logic multiplication table using the & operator.



Please note the result of (TRUE | NA). It is TRUE because regardless of the fact whether the indeterminate value turns out to be true or false, its logic sum with the TRUE value will yield the TRUE value. Accordingly, FALSE & NA returns the value of FALSE.

In the R program, there are also the longer forms of the operators | and &, i.e. || and &&. There are two differences between the longer and shorter form. The shorter form operates on the vectors and performs the vector element operations one by one, whilst the longer form only operates on the first elements of the vectors, and returns the one-element value of TRUE or FALSE. Another difference is related to the advanced applications. The longer form does evaluate the right argument only if it is necessary.

For example, the first line of the below example will execute correctly, as there is no need to evaluate the function cat () to determine the result. The second example will execute the function cat () and indicate an error, because it is not possible to logically add the function cat() to the value TRUE.



**Data Downloading**

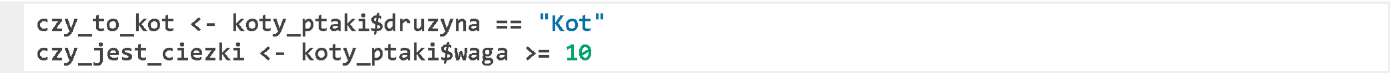
The data we start our examples with include koty\_ptaki of the PogromcyDanych package. To download these data, please run the package (for instructions how to do it, see Episode 2).

Let’s run the package and use the function head() to display first six rows.



**Descriptive Statistics**

On the basis of the variable druzyna, we will build a logic variable czy\_to\_kot, and on the basis of the variable waga, we will build a logic variable czy\_jest\_ciezki.



If the logic variable is considered a bivalent qualitative variable, we can apply, for example, a frequency table function (one- and two-dimensional) to such variable.

The results of the below frequency table for two variables show that there is only one animal in this data set, which is both heavier than 10 kg and is not a cat.



**Negation Operator**

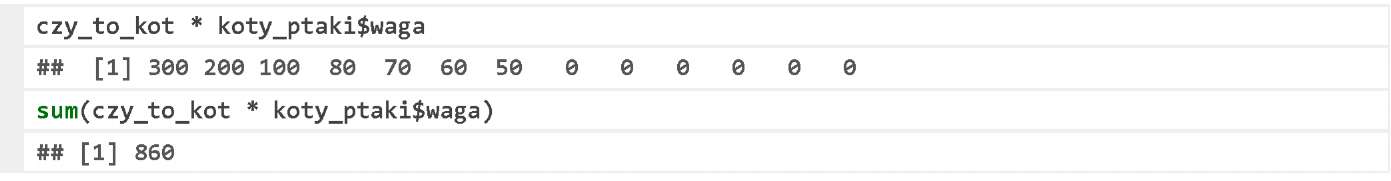
Another useful operator is a logic negation !. It converts the logic truth value into falsehood value and vice versa.



In the arithmetic operations, the TRUE logic values are converted to 1, and the FALSE logic value to 0. Therefore, if we want to calculate how many values are TRUE, all you need to do is to sum up the vector value. The example below shows how many cats and no-cats are in the set of data.



Such a logic value vector can be multiplied by the numerical value vector. The instruction below uses such an operator to calculate the total weight of all cats in the data set. The multiplication by the logic value vector results in resetting some of the values.

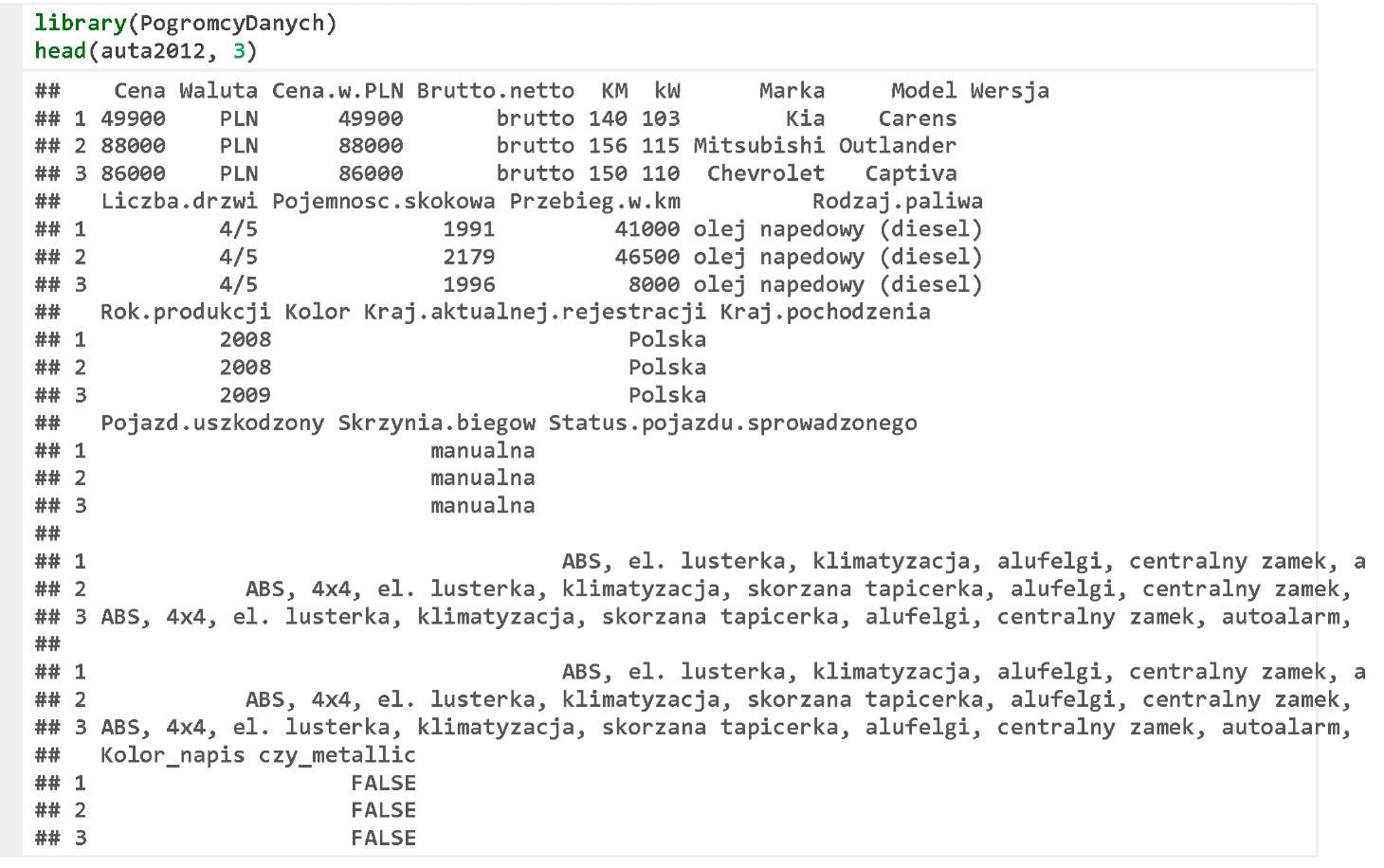


**Data Downloading**

The data set named koty\_ptaki is made up of 13 rows. The whole data set can be displayed on the screen. We not always need a special descriptive statistics to understand what is going on in such a small data set.

Therefore, the next exercises on the logic variables will be done using a considerably larger set of data auta2012 including more than 200,000 values, which is also contained in the PogromcyDanych package.

The description of this data set can be found in the Episode <https://rawgit.com/pbiecek/MOOC/master/0_dane/0_dane.html>. Let’s download this set of data and analyse the first three rows.



**Descriptive Statistics**

Let’s use the logic values and the function table() to check how many cars in our data set have been registered in Poland and how many of them had been manufactured before 2007. Similarly as for the qualitative data, we can use the function prop.table() to determine the fractions/percentages.

Surprisingly, there are less than half of the cars for which the declared country of current registration is Poland.



**Exercises:**

Check the offer price of how many cars is below PLN 2,000.

Check what the share of the cars with the engine capacity in excess of 1,500 cm3 is.

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

**Type and Value Tests**

Please, in comparing the values pay a special attention to the indeterminate values, which sometimes behave non-intuitively, but always logically.

For example, the operator ! = tests whether the values are different (it is a negation of the operator ==). How to check whether the vector values are different than the missing NA value? Let’s start with an example how not to do it.



The result of the != NA comparison is the value vector of NA. It is so because if the right side of the comparison is unknown, we do not know whether the value of 41,000 is different than others or not.

To check whether a specific value is a missing value, we can use the function of is.na(). The result is TRUE if the tested value is NA, and FALSE if it is not.

We can use this function to calculate the percentage of the values for which no data are missing.



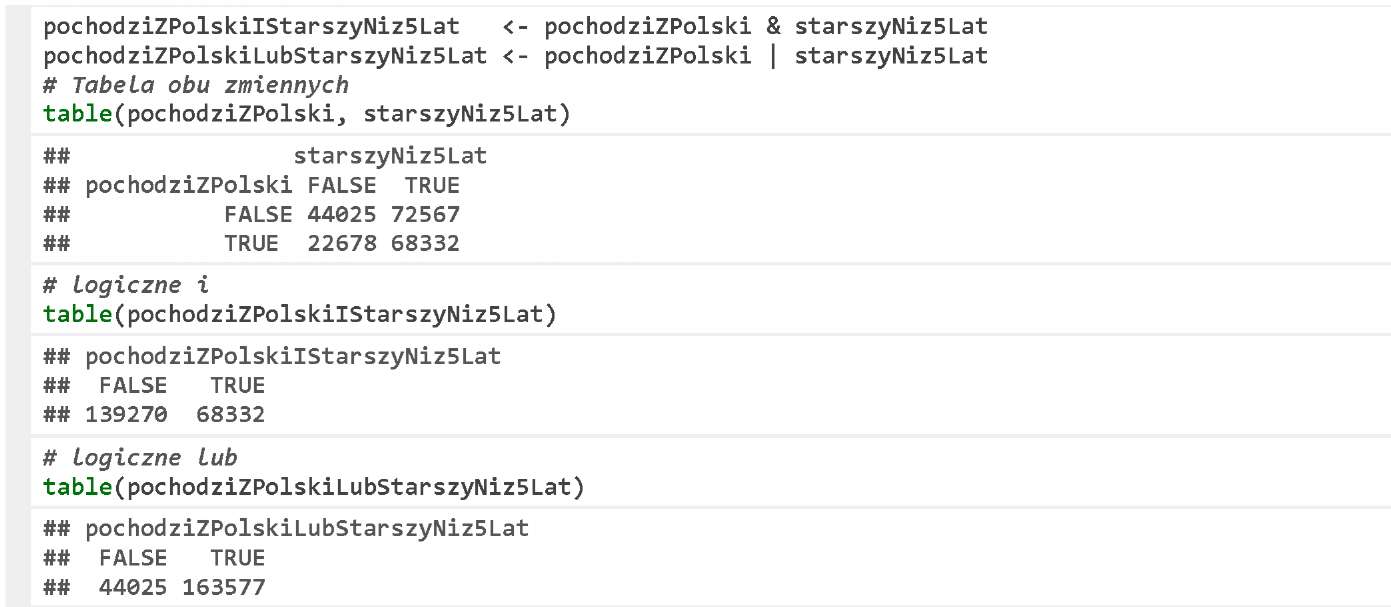
There are more test functions. Their names start with is., so all you need is to enter three characters in the console and press TAB to see a list of them. Some of them check whether a variable is a specific type variable, e.g. is. factor(), is. numeric(), and some check whether a variable contains specific values, e.g. is.na() or is.nan().

**Logical ‘and’ and logical ‘or’**

Since the logic operators or and and are commonly used, we will practise them once again using the car data set. Let’s define two logic value vectors, which verify whether a car is presently registered in Poland, and whether it is older than five years, i.e. manufactured before 2007.



Now, let’s calculate the logic sum and the product of both vectors. Then, we will use the function table to present the results.



**Exercises:**

* Check the offer price of how many cars registered in Poland is below PLN 2,000.
* Check what the share of the cars with the engine capacity in excess of 1,500 cm3 and using fuel oil is.

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