**String of Characters Features**

pogRomcy danych (Data Masters) Season 1 / Episode 14

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. They very often include the string of characters or the variables to be converted to the string of characters (e.g. date, which can be converted to a string of characters to extract year and month).

In this episode, we will learn:

* How to convert the variables from and into the strings of characters.
* What basic operations can be performed on the strings of characters.
* How to sum up/describe the strings of characters.

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 is one auta2012, both available in the PogromcyDanych package.

**What Does String of Characters Mean?**

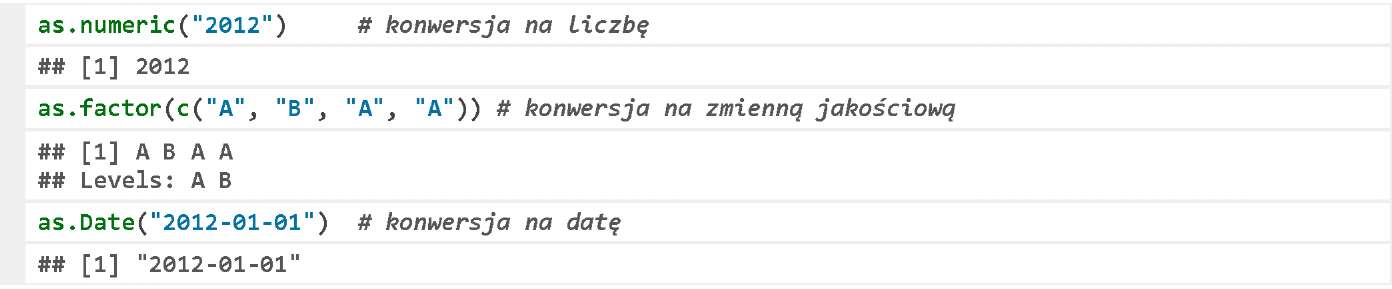
The persons experienced in working with the languages, such as C++ or Java are accustomed to the following type classification: characters, strings of characters, and vectors of strings of characters. There is no such classification in the R program, except the vectors of strings of characters.

A vector can be made up of one element, and it can be one element long, but still it is a vector of strings of characters. Such vectors are marked as the character class.



The strings of characters can be found in various contexts. The most natural is that a set of data contains the statements collected. For example, when retrieving the data from Twitter, one of the variables, which describe a single “twit”, is its content, i.e. a string of 140 characters.

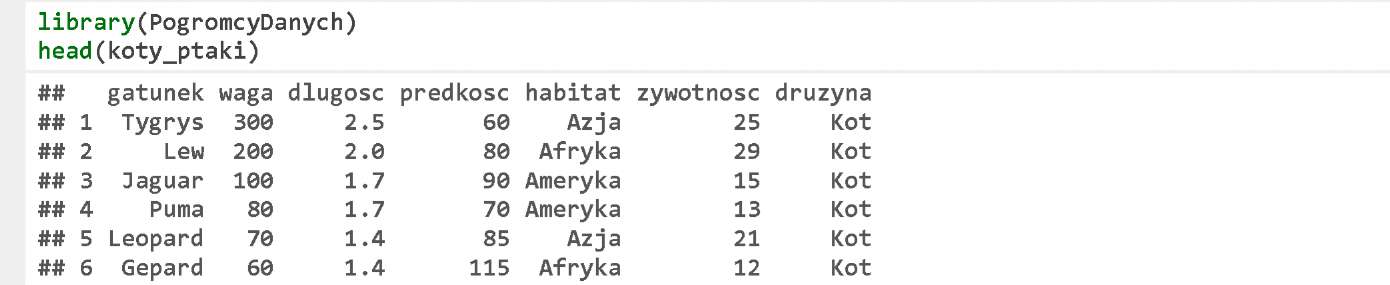
The strings of characters can also be found when downloading the quantitative or qualitative data, if such data, for any reasons whatsoever, have been interpreted as the strings of characters in the course of the downloading process (e.g. due to an improperly defined formatting). In such a case, we often convert them into a desired type, such as quantitative, qualitative, date, etc.



**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.



Each row describes one species. As we know from the previous episode, the variables species, habitat and druzyna are the qualitative variables. We will convert them into the strings of characters to demonstrate the basic operations with the strings of characters.

**Conversion into String of Characters**

Select the column habitat. It is a qualitative variable in the koty\_ptaki set of data. Let’s use the function as.character() to convert it into a string of characters. We will save the resulting strings of characters in the column habitat\_napis.

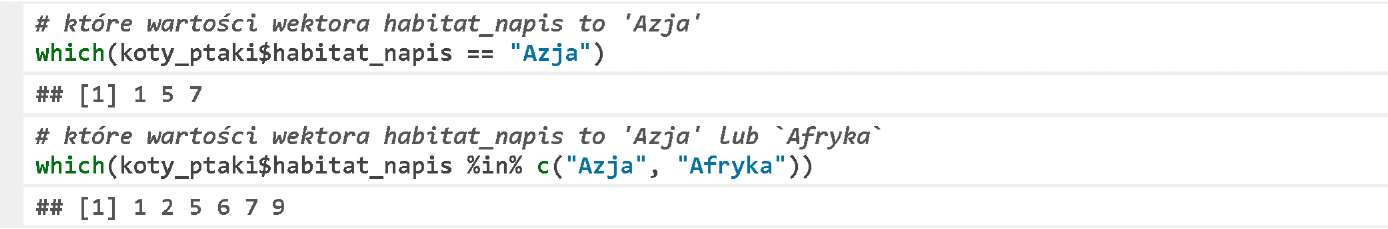


The function length() checks the length of each vector. The function nchar() checks the number of the characters in each string.



**String of Characters Retrieval**

A common string of characters operation is to search for the strings of characters that match a pattern. If a character accuracy is important for us, the use of the function which() would be convenient. The result returns the indexes, for which a specified condition is true.



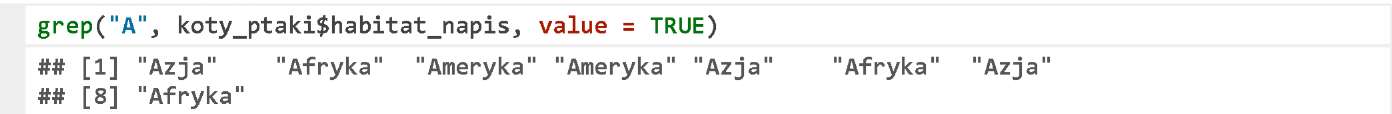
However, it is often more convenient to describe the values to be retrieved using a certain pattern rather than to list all of them. The function grep() is used for such retrieval. It tests, which strings of characters contain a specific string of characters, and, as a result, returns the numbers of the strings of characters with such string.

Let’s check, which rows in the column habitat\_napis include A character in its name. The first argument of the function grep() is a pattern, the second – the vector of strings of characters. We can specify these arguments in a different order, but then we will need to specify their name.

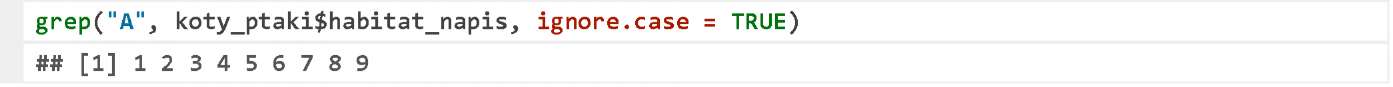


**String of Characters Retrieval**

If instead of the indexes, we would like to get the values, which were matched, then we should set the argument value as TRUE.



Another useful argument of the function grep() is a possibility to define that the search should be performed disregarding information whether the character is small or capital. We use the argument ignore. case=TRUE to do so.



In defining a pattern, we can use the regular expressions. They allow describing a certain regularity to be searched for in the data. The regular expressions are often used to check whether a string of characters is, for example, an email, a telephone number, or a zip code.

For more details concerning the regular expressions, see, for example, Wikipedia at http://pl.wikipedia.org/wiki/Wyra%C5%BCenie\_regularne#Wyra.C5.BCenia\_regularne\_w\_praktyce. The below expression checks whether the string of characters starts with A or E.



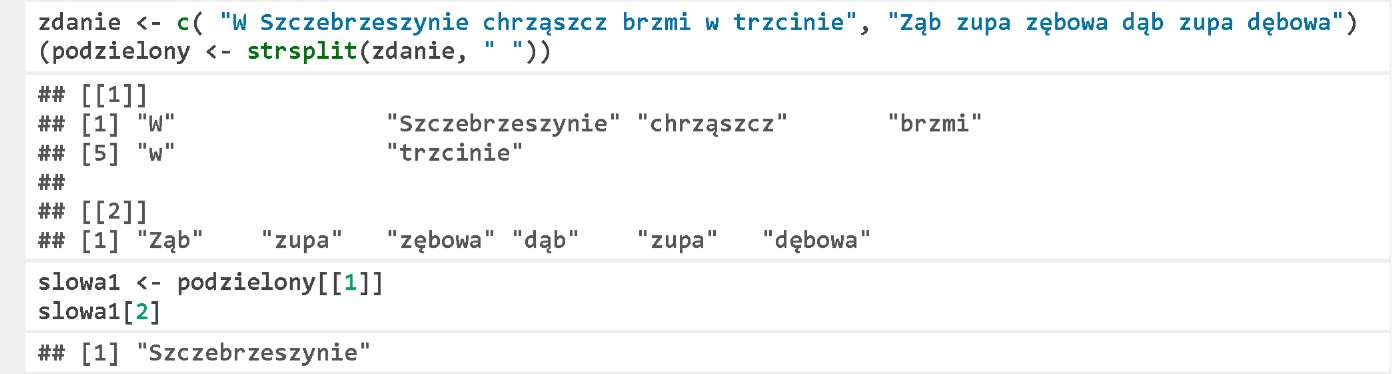
**String of Characters Fragment**

Another useful feature for operating with the strings of characters is substr(). It enables to cut out a fragment at a specific position from a string of characters; the second and the third arguments are the indexes of the beginning and end of the relevant string of characters.

For example, in this way we can remove years or months from the dates.



Another method to retrieve a string of characters element is to define a pattern, which separates the important elements of a string of characters, such as words. The below example features two sentences broken into the words (specifically, broken into the fragments separated with a space). It yields a two-element list of the vectors. We select the first vector, and read its second word.



**Exercise**

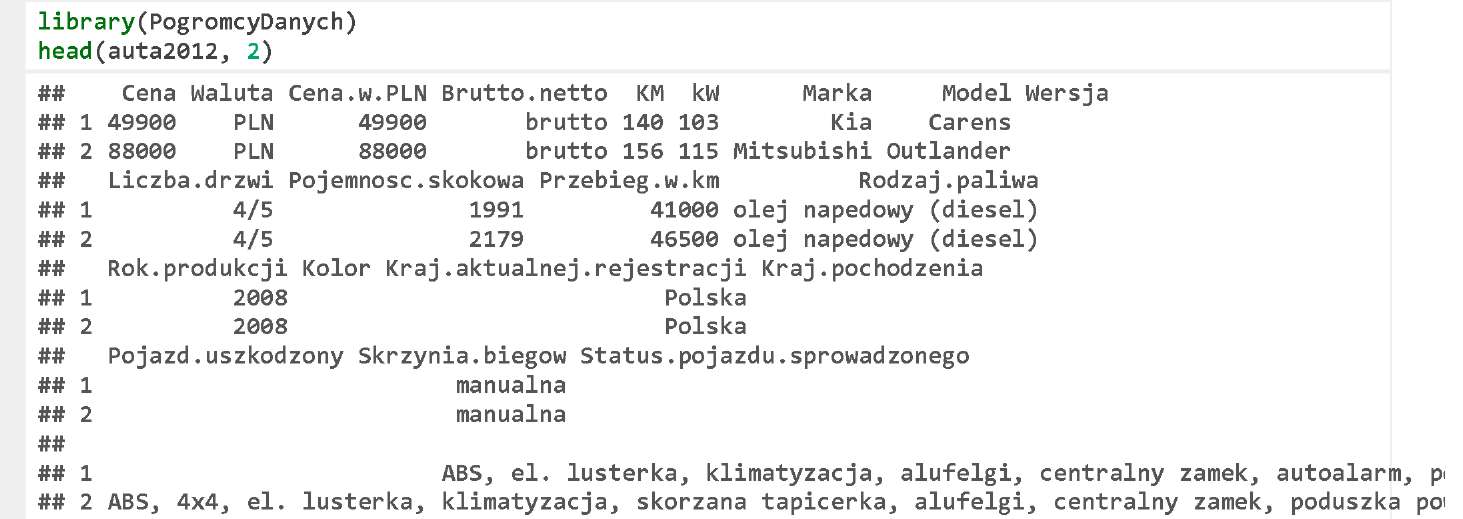
* Find the row indexes in the set of data koty\_ptaki, in which gatunek contains a small or capital letter S.
* Find all rows in which any species of a falcon is described.

**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 quantitative 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/O\_dane/O\_dane.html.Let’s download this set of data and analyse the first two rows.



Two columns Wyposażenie. dodatkowe and Kolor will be of a special interest for us as far as the analysis of the strings of characters is concerned.

**Strings of Characters**

When downloading the data to the R program, most of the functions replace the strings of characters with the qualitative variables by default. This issue has been more widely described in the Episode 5.

If we want to process the strings of characters, we need to convert the qualitative variables into the strings of characters using the feature as.character().

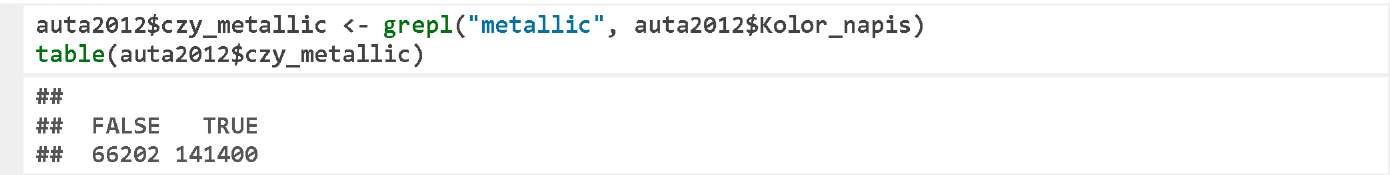


**Strings of Characters**

In the data analysis, we often face the need to create the new features on the basis of the old ones. For example, if we look at the car colour vector, we could consider a question whether the lacquer paint is metallic or not be an interesting feature.



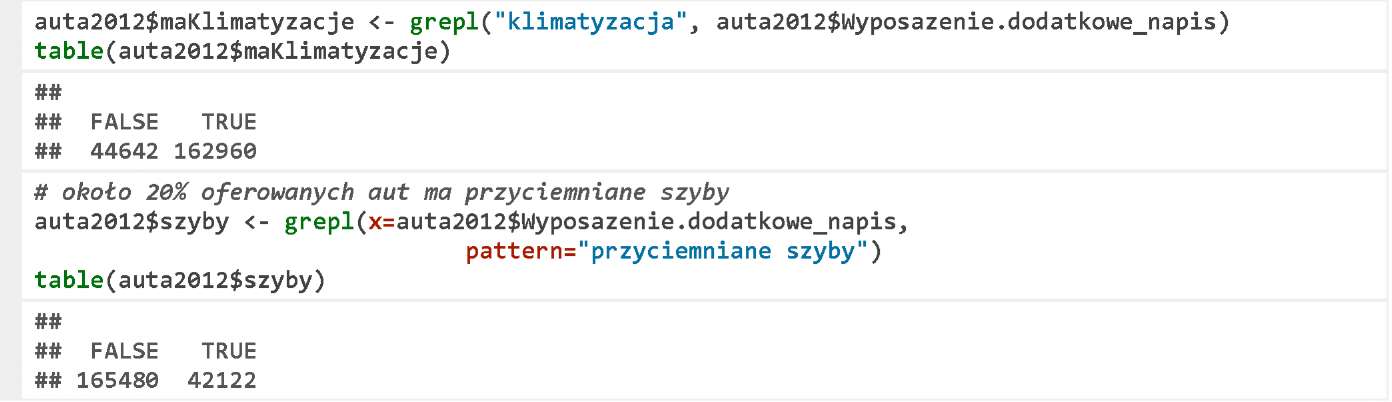
You can do it in several ways. The easiest way is to use the function grepl(), with the same parameters as the function grep() except that the result is the logic value vector TRUE/FALSE stating whether a given pattern has been found in the strings of characters.



**Strings of Characters**

In a similar way, you can create the additional columns that determine whether a car is equipped with air conditioning, or the windows are tinted.

In the second example, it is worth noting that the arguments are listed in the order other than the default order. Therefore, a specific name pattern= was required for the argument with a pattern.

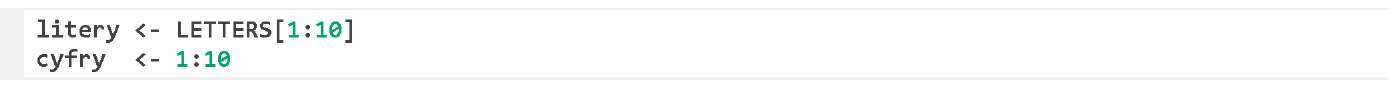


**Strings of Characters Pasting**

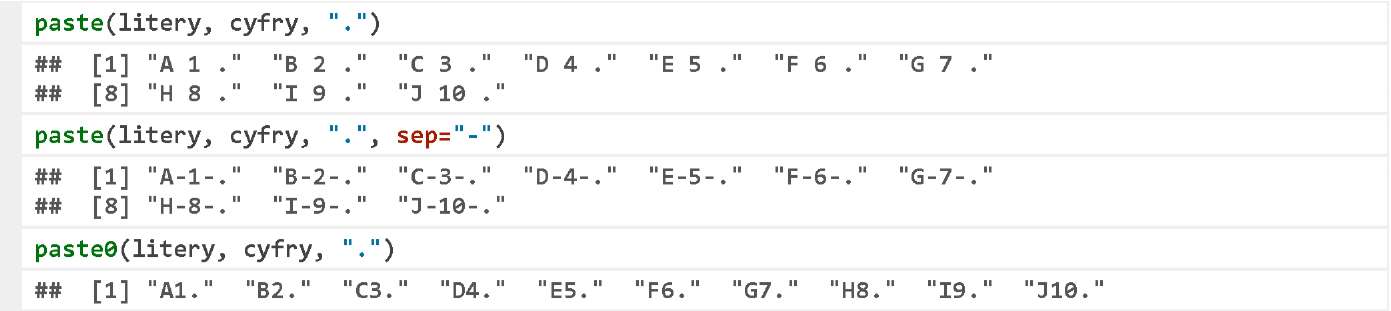
An operation that is opposite to the string of characters cutting out into the fragments, is an operation of pasting several strings of characters to form one string of characters. For this purpose, you can use the function paste() or paste0().

We will present its effects in the below, simple example. We will paste a vector including ten letters, ten numbers, and one string of characters.

If the functions in the R program, which operate on the vectors (most of them), include two arguments of different length, the shorter argument will be multiplied as many times as needed to make its length equal the length of the longer one (so-called recycling rule). For this reason, the third argument – a single dot – will be multiplied and pasted to each of ten elements of the vector litery and cyfry.



The function paste() pastes the values while separating them with the separators, i.e. a space by default. You can change a separator character by indicating the argument sep. The function paste0() differs from the function paste() in such a way that the separator is an empty string of characters.

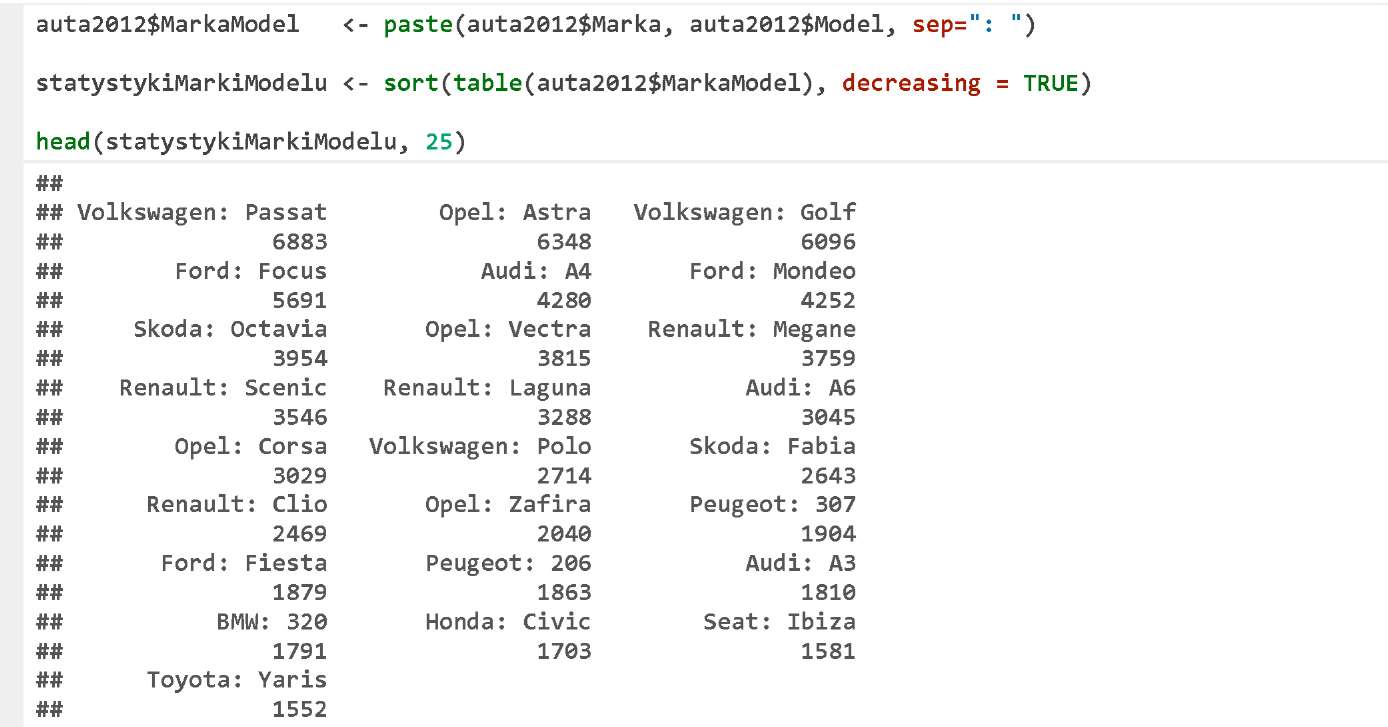


**Strings of Characters Pasting**

Let’s present the paste function illustrated with an example of pasting a name of a car make and model.

We will paste both vectors in the first row considering ': ' as the separator. Next, we will count the frequency of such pairs and put them in the descending order.

Please find below 25 most common models.



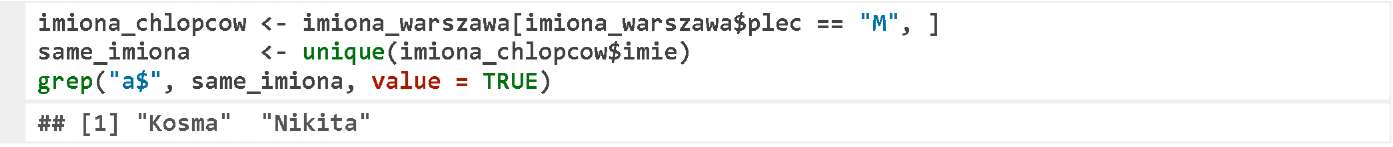
**Names of Children**

The package PogromcyDanych also contains the set of data named imiona\_warszawa including the popularity statistics of the names given to the new-borns in Warsaw in different periods.

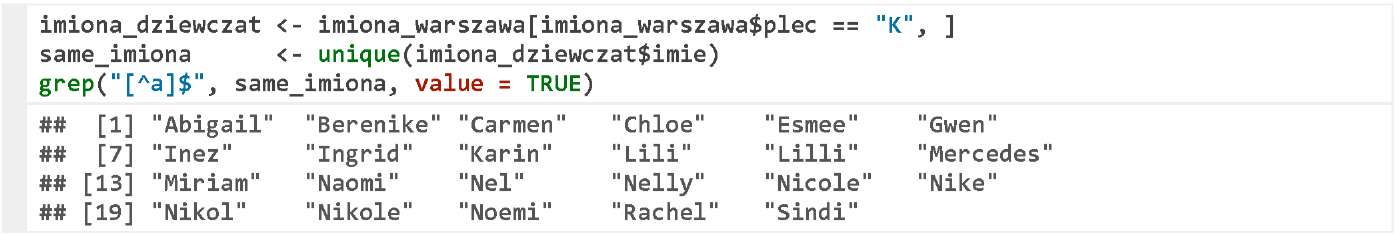
Using this set of data we can test a rule that a female name ends with ‘a’, and a male name does not. Some time ago, the only exception was a male name Bonawentura. Has anything changed?

Let’s select only the male names to be analysed (indexing imiona\_warszawa$plec == "M"). Since each name is repeated for each month, we use the function unique() to remove all duplicates.

Next, using the function grep() we will select all names with the last letter of a (regular expression a$), and display these names. There is no Bonawentura, but there are other names.



We will repeat a similar operation for the girls. This time, we will determine all letters except a, i.e. [Aa]$ for a regular expression pattern. There are many such female names.



**Numbers, Qualitative Features, and Strings of Characters**

The function as.character() can be used also if we want to convert a qualitative variable into a quantitative variable. We are not allowed to do it directly, but using the function as. character().

Let’s demonstrate this problem with an example. We will convert the variables into the qualitative variables using the function as.factor(), and next into the numbers with the function as.numeric(). It turns out, however, that the result is not as expected.



The qualitative variables are converted using the function as. numeric() into the subsequent integers. If we want to retrieve the numbers being a part of the factors, we must convert them into the strings of characters first using the function as. character().



**Exercises:**

Check how many cars are equipped with the alarm system (referred to as ‘autoalarm’).

Check the lacquer paint of how many cars is metallic (referred to as ‘metallic’).

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

**Where Can You Find More Information**

We have only outlined a topic of the strings of characters analysis.

For processing larger text corpuses, many interesting topics come up, such as the analysis of meaning, the frequency analysis, tagging, etc.

You can find more details on the basic operations with the strings of characters in the chapter 2.1 of the “R Package Guide” GiS 2013. This chapter is available free of charge at http://biecek.pl/R/.

When working with the large corpuses, you can accelerate the computations using the package stringr. The very well developed material concerning the features of this package is shared by Gaston Sanchez at http://gastonsanchez.com/Handling\_and\_Processing\_Strings\_in\_R.pdf.

When working with the strings of characters, the expressions and metacharacters are very useful. A brief, but concise overview of the regular expressions developed by S. Jones is included in the document “String Manipulation in R” at http://www3.nd.edu/~sjones20/JonesUND/BioStats\_files/RstringManipulation\_2-6-13.pdf.

A very interesting alternative that supports more exotic languages (more exotic than English) is the package stringi. You can listen to the speech of Marek Gągolewski at the 2nd Meeting of the R Enthusiasts. The video is available at http://PogromcyDanych.pl/SER/#SERII.