Problem Set 10

Andreas Bender, Philipp Kopper, Sven Lorenz

30 January 2022

Read chapter 14 on strings in R4DS. Skip the exercises. Read this website post, too. Focus on character classes. Next to the functions introduced on the slides, you may probably need to use:

- `str_to_lower`
- `str_split`
- `str_remove`
- `str_trim`
- `str_to_title`

Furthermore, you will need some relatively complex regular expressions. Use the mattermost channel to exchange with your fellow students on these regular expressions.

Like in the weeks before, we provide solutions for you to check your own approach. This week you have to set up the testing yourself, though, if you wish to test your own solutions. To recycle your code from the last few weeks, you may follow the convention to name your solutions accourding to the subproblem it refers to (e.g. `ex1a`).

Exercise 1

On the Moodle page there is a newspaper article which should be prepared for the use in a machine learning algorithm. So far, the article is just like it was scraped from the web page.

a) The algorithm cannot deal with special characters except ".", and ",". Remove all other special characters. Additionally, the text should be converted to lower cases only. Make sure that these special characters are conserved. This means you should avoid the deletion of special characters which are either directly before or after numbers.

You solution should look like this:

```
## [1] "der von der griechischen justiz verfolgte frühere "
```

b) Change the German Umlaute (e.g. ä) into the international equivalent (e.g. ae).

c) On Moodle, we also provide a vector with the 50 most frequently used words in German. Remove all words occurring in this vector from the text. Only remove the words reported in the vector and no variants. (As you may have seen the vector has more than 50 entries. We already added some variations of the most frequent German words.)

Exercise 2

For this exercise you need to download the csv dataset `adressliste.csv` from the Moodle page of this course. Use the `stringr` package to solve this task.

a) Read in the dataset `adressliste.csv` and think of a sensible name. In each subtask please overwrite the original dataset and keep the original name throughout the whole exercise. The same applies for all columns. After reading in the file, change all German characters (e.g. ä, ü, ö and ß) in all columns that contain any into their international equivalents (e.g. ae and so on).

b) Change the column `Adresse` such that it contains the abbreviation `str.` instead of "Strasse"/"Straße" for all addresses of concern.

c) The columns `Wohnort` and `PLZ` are a bit messed up. Sometimes `Wohnort` contains the city only and sometimes it also contains the postal code. Change the two columns such that the column `Wohnort` contains the city only (e.g. Muenchen) and the column `PLZ` contains the postal code. After doing this, there should be no `NA`s left in the `PLZ` column anymore.

d) The column `Geburtsdatum` contains the date of birth for each person. However, the dates are not consistent. change them such that each date looks like this `03.09.1984`. After doing that transformation, change the column `Geburtsdatum` such that it is of class `Date`. That way, **R** also understands that it is a date column. From this, calculate a new column `Age` that contains the age in years for each person in the dataset.

e) What is the mean age of people whose first name start or end with a vowel? What is the address of the people whose last name has any two identical letters in a row?

f) The column `Nummer` contains the phone number of each person. Again, they are not consistent and some are also missing. Some numbers start with 089 for munich, some start with 89 and some do not start with either, so they are just the phone number without the munich area code. Change all telephone numbers such that they start with +49 followed by the area code and then the number. For example: +49 89 1234 (mind the blanks!).

g) For each of the following specifications, extract the data set that only contains : Phone numbers with at least two or more of the same digit in a row Phone numbers with a pair of numbers followed by the same numbers in reversed order (e.g. 6776) Phone numbers that start and end with the same number, do not take the "+49 89" part into account.

h) In a final step change the names, addresses and cities such that they start with a capital letter. Now your dataset should look nicely.