Sprachverarbeitung: Übung

SoSe 24

Janis Pagel
Department for Digital Humanities, University of Cologne

2024-04-16

Please submit your solution via Ilias, either as a Jupyter Notebook (`.ipynb`, you can export your Notebook in Jupyter by going to File > Download) or as a Python script (`.py`) if you are not working in Jupyter. You do not need to submit the output file `chars.tsv` or the plot of the last exercise, only the Python code that generates these outputs.

Exercise 1.

Download the file `romeo.txt` from `https://lehre.idh.uni-koeln.de/site/assets/files/5151/romeo.txt` and write a function `read_file()` in Python takes as an argument a path to a file, reads in that file and returns the file contents as a string. Write a second function `count_chars()` that counts all the characters (except for newline `"\n"` and tab `"\t"`) in a given string and returns the result as a dictionary with the characters as the keys and their counts as values. Apply it to the output of `read_file()`. Write a third function `write_chars_file()` that takes the output of `count_chars()` as input and writes it into a file called `chars.tsv` with the following format:

- First line: Header names with two columns separated by a tab (the string symbol for the tab in Python is `"\t"`), called "character" and "count"
- Subsequent lines: the characters and their counts, one character and count per line, separated with a tab

so that your output file looks something like this:

```
character	count
a	34
b	2
c	500
...
```

(These counts are made up examples and the order of the characters in your resulting file is not important)

Hint 1: In order to exclude newlines and tabs in the function `count_chars()`, you can use nested if statements and the "not-equal-to" operator `!=`, for example `if x != "a":` means that the if statement is only true if the value of `x` is not `"a"`

Hint 2: For the function `write_chars_file()` it might make sense to first convert the input dictionary into a list of strings with a single line per item and then write the contents of this list into a file.

Solution 1.

```
def read_file(path):
    with open(path, "r") as fo:
        fc = fo.read()
    return fc

def count_chars(string):
    char_dict = {}
    for char in string:
        if char != "\n":
            if char != "\t":
                if char in char_dict:
                    char_dict[char] += 1
                else:
                    char_dict[char] = 1
    return char_dict

def write_chars_file(char_dict):
    lines = []
    for char in char_dict:
        lines.append(f"{char}\t{char_dict[char]}")
    with open("chars.tsv", "w") as fo:
        fo.write("character\tcount\n")
        fo.write("\n".join(lines))
```

Exercise 2.

Read the file from the previous exercise (`chars.tsv`) into a pandas dataframe (if you did not solve Exercise 1, you can download and use the file under the following link: `https://lehre.idh.uni-koeln.de/site/assets/files/5151/chars.tsv`). The `read_csv` function of pandas can read tab-separated data (`tsv`), inform yourself online how to read such a file with `read_csv`.

Add a new column to the dataframe called "alpha", that contains the value `True` if the character is a letter and `False` if the character is not a letter. To solve this task, inform yourself about the pandas methods "isalpha()" (`https://pandas.pydata.org/docs/reference/api/pandas.Series.str.isalpha.html`) and how to use it. Also inform yourself about how to add the output of "isalpha()" to a new column in the dataframe.

Finally, produce a new dataframe that contains the mean counts for two groups: "al-

pha is True" and "alpha is False".

Hint 1: You have to set `quoting=3` as a parameter for the `read_csv` function of pandas in order to handle the double and single quotes in the data.

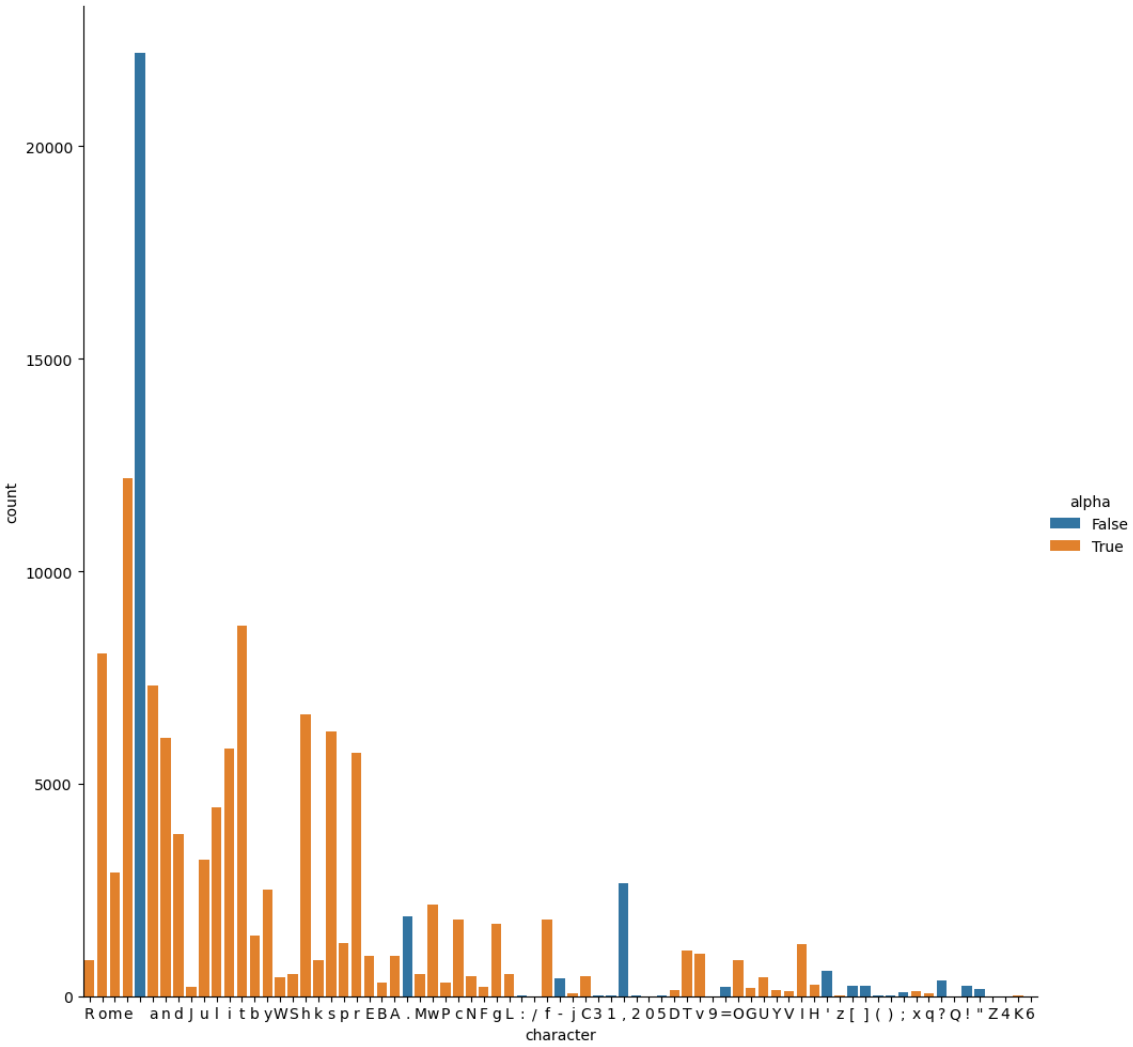
Hint 2: In order to create the dataframe with the mean values, you only need the columns "count" and "alpha", not the column "character"

Solution 2.

```
import pandas as pd
df = pd.read_csv("chars.tsv", sep="\t", quoting=3, header=0)
print(df)
df["alpha"] = df.character.str.isalpha()
print(df)
print(df[["alpha", "count"]].groupby("alpha").mean())
```

Exercise 3.

Use the dataframe from the previous exercise (with the added alpha column) and create a barplot with seaborn's "catplot()" function that has each character on the x-axis, its count on the y axis and is colored according to the value in the "alpha" column. Adjust the size of the resulting plot to your liking in case it is too small, via the `height=value` parameter of "catplot()". Your resulting plot should look something like this:



Solution 3.

```
import seaborn as sns
sns.catplot(kind = "bar", data=df, x = "character", y = "count", hue = "
                                    alpha", height = 10)
```