Exercise Week 6 – Reference Solution

Sprachverarbeitung (VL + Ü)

Nils Reiter, nils.reiter@uni-koeln.de

May 9, 2023 (Summer term 2023)

The file `/teaching/summer-2023/sprachverarbeitung/data/titanic.csv` contains passenger data from the crossing of the Titanic, which sank in the North Atlantic on 15 April 1912.[1] The file contains information on the passengers, together with details of who survived the sinking and who did not.

The following tasks require a mixture between reading/thinking and applying command line tools. All tasks can be solved purely manual, but this is cumbersome and takes a lot of time.

1. Find out what features are given for the passengers.
   - See below
2. Assign a data type to each feature.
   - PassengerId: numeric, Survived: categorial, Pclass: categorial, Name: string, Sex: categorial, Age: numeric, SibSp (number of siblings and spouses): numeric, Parch (number of parents and children): numeric, Ticket: string, Fare: numeric, Cabin: string, Embarked: categorial
3. For category features: Find out which categories there are, and how common each is.
   - The following strategy lets us extract the values of one column:
     a) We first remove all double double quotes (i.e., occurrences of two adjacent symbols `"`). They are sometimes used to mark nick names in the name column.
     `sed -E 's/""//g'`
     b) We then fix the name column, mostly by removing the comma separating last and first name. The core idea in the RE is to look for something that is enclosed by `,"` on the left, `",` on the right, and has a comma in the middle. We group the stuff left and right of the comma, and insert it without comma.
     `sed -E "s/,\"([[:alpha:] '-]+?), ([[:alpha:]. ()\/-]+?)\",/,\1 \2,/g"`

[1] `https://en.wikipedia.org/wiki/Titanic`

c) We now have a clean table, with commas purely used as separators. Thus, we can use an earlier idea to isolate a single column (e.g., column n), by removing the first $n-1$ columns, including their commas. Now we have column n at the beginning.
`sed -E 's/^([^,]*,){5}//g'` (for column $n = 6$)

d) Now we remove everything from the first comma to the end.
`sed -E "s/,.*$//g"`

e) Finally, we have a single column, and can use the sort / uniq combination as before. This will print us how many of each values we have in the table.
`sort | uniq -c`

- The full pipeline looks like this (you need to type it in one row):

```
cat titanic.csv
  | sed -E 's/""//g'
  | sed -E "s/,\"([[:alpha:] '-]+?), ([[:alpha:]. ()\/-]+?)\",/,\1 \2,/g"
  | sed -E 's/^([^,]*,){1}//g'
  | sed -E "s/,.*$//g"
  | sort | uniq -c
```

4. For numeric features: Find out what range of values they assume (minimum and maximum).
   - We can apply the same pipeline as above to isolate a column of numbers. Then we can use `sort -n` to get it sorted numerically, and we can inspect the top and bottom of the list.

Hints

- One column is particularly tricky, because it contains relatively free text, including commas and quotation marks. Maybe first clean up this column by removing commas etc. or remove the column entirely. Both can be done with regular expressions as we have used them before.
- Accessing individual columns could be done with a similar trick as when we defined the context for the concordance in words instead of characters: Define some content of the column and the character that separates the columns as a group, and ask this group to be repeated.