Assignment 2 OpenRefine (Airbnb dataset)

Backstory. Your family is visiting you in Illinois for the very first time, and you decide to take them to Chicago for a short trip. You wish to give them the best Chicago experience, but the hotels in Chicago are just way beyond your budget. Instead, you decide to stay at a Bed & Breakfast (BnB). You know that in order to choose a perfect BnB, you have to scrutinize and carefully inspect the listings. Therefore, you gather the latest Airbnb Chicago listing dataset, and start to put your OpenRefine skills that you've learned in class into practice.

Your ultimate goal is to: clean the dataset to a certain, acceptable level, so that it is good to use for further data analysis (not just for your family).

STEP 1: (if you haven't yet:) Download and install OpenRefine **3.7.2**
https://openrefine.org/download.html
Note: If you are a Windows user, use the version "Windows (including Java)".

STEP 2: Create Project → Load the **airbnb_dirty.csv** into OpenRefine. Make sure it is loaded in CSV format. **Uncheck** the default option '**Trim leading & trailing whitespace from strings**'. (In order to record all cleaning steps in the OpenRefine operation history/recipe, we ask you to trim whitespaces not automatically on import, but rather explicitly in the UI.)

STEP 3: Complete the data cleaning tasks below.

***IMPORTANT NOTE**:

- For the purpose of grading and track changes, do **NOT** do any edits on the ***ids*** column
- Do **NOT** open the **airbnb_dirty.csv** file in any other application (e.g. MS Excel) as these applications might do some unintended character conversions and your final "clean dataset" may have some non-standard characters introduced by the application.
- Execute the tasks in sequential order, step by step. Do **NOT** skip any steps.

1. **TRIM and COLLAPSE WHITESPACE CHARACTERS**.
   - Description: It is very common to see extraneous whitespace characters in datasets. Often whitespaces are hidden at the beginning or the end of a string, and sometimes they are hidden as two or more consecutive whitespaces between words. Here is what you can do to clean up (or *trim*) such redundant whitespace characters.
   - Tasks:
     - **Trim all the leading and trailing whitespaces** in the following columns of type text (string): **name**, **host_name**, **neighbourhood**, and **room_type**.
     - **Collapse consecutive whitespaces** in the same four columns.

2. **TYPE CONVERSION to NUMBER**.
   - Description: If a column C contains numeric values but is of type text (string), before applying numeric operations, we need to convert C to type "Number" in OpenRefine.
   - Tasks:
     - Transform the following columns "to number": **host_id, latitude, longitude, price, mininum_nights, number_of_reviews, reviews_per_month, calculated_host_listings_count**, and **availaibility_365**..
     - Note that numeric columns will be shown in green.

3. **CONVERSION to Titlecase, UPPERCASE, lowercase**
   - Description: Sometimes you want to convert string values to lowercase, UPPERCASE (a.k.a. ALL CAPS), or Title Case. When you're going through the Airbnb dataset, you will notice that most of the values in the **neighbourhood** column are using Title Case (e.g. **L**ogan **S**quare), but some are not.
   - Tasks:
     - **Add** a new column based on the **neighbourhood** column, name the new column as **neighbourhood_case**.
       - **Note the (British) spelling of neighbourhood_case!**
     - Transform the **neighbourhood_case** column to **Title Case**.

4. **MASS EDITS with FACETS**.
   - Description: Faceting[1] is a powerful feature of OpenRefine that allows you to interactively explore and "profile" a dataset. It can also be used to "bulk execute" many edits with a single operation (i.e., to perform a *mass edit*) in order to obtain canonical string representations. Depending on the data type, one can create numeric facets, text facets, or scatterplot facets.
   - Tasks:
     - **Add** a new column based on the **neighbourhood_case** and name the new column as **neighbourhood_loop** (*not* **neighborhood_loop)**
     - Using the **neighbourhood_loop** column, create a **text facet** and sort by **count**!

[1] https://openrefine.org/docs/manual/facets

- In the original dataset, the string “L??p” appears over 300 times. Replace the question mark “?” character[2] by changing “**L??p”** to “**Loop”** using facets.
- Close (remove) the **neighbourhood_loop** text fact after completing this step.

5. **CLUSTERING.**
   - Description: Clustering is used to group similar items together. In data cleaning we often want to *canonicalize* similar string values, i.e., if they are the result of spelling variations and typos. OpenRefine allows clustering using different methods and key functions.
   - Tasks:
     - **Add** a new column based on the **neighbourhood_loop** column, and name the new column as **neighbourhood_cluster**.
     - Using the **neighbourhood_cluster** column, create a text facet.
     - On the **text facet** box for **neighbourhood_cluster**, click **Cluster**.
     - You’ll immediately see several different spellings of **OHare**. Use the spelling “O**'H**are” (i.e., the one with the apostrophe[3] character: **'**) as the new cell value by checking the box for **Merge?** and then clicking **Merge Selected and Recluster**.
     - Experiment with different combinations of *Method* and *Key functions* and fix other clusters. **Note:** In addition to the cluster of values that map to “O'Hare”, there should only be one additional cluster with different spellings (or typos) of “West Garfield Park”. Make sure to use “**West Garfield Park**” as the canonical name while **avoiding** to change other values (e.g., “East Garfield Park” is a *false positive*: if it’s in a cluster together with “West Garfield Park”, **do not merge!**).
     - Make sure to **close** (remove) the **neighbourhood_cluster** text facet window after you are done with this task.

6. **SPLIT COLUMNS.**
   - Description: In the **host_name** column, many cells include two (or more) people’s names joined by “**And**”. For instance, there’s an instance of “Michael And Veronica”.
   - Tasks:
     - **Split** these joint host names into separate columns so each of the cells only contains one name.
     - However, you should not split names such as **And**rea, **And**y, or **And**rew.
     - To achieve this, you will need to use **regular expressions** when splitting columns (remember to tick the ‘regular expression’ box).
     - Also keep the original **host_name** column (**uncheck** the ‘*remove this column*’ box in the split column window).

---

[2] Such question marks may be the result of unintended conversions of *diacritic* characters, e.g., when an application replaced the string “Lóóp” with “L??p”.

[3] Quotes have different typographies. There is an apostrophe (') which is an ASCII character but there are open single quote (‘) and closing single quote (’) too which are not ASCII. The original dataset had all three of these. For your convenience, we have replaced all non-ASCII quotes with “?” in the dataset, so you will also see “O?hare” as one of the spellings. You can learn more about the different typography of quotes here.

7. **DELETE EMPTY COLUMNS.** You notice that the **neighbourhood_group** column has no values, so you decide that this column should be deleted. **Delete** this column!

8. **CONVERSION TO DATE**. The **last_review** column apparently is in a date format: Transform it into ISO standard date form!

9. **WORKING with GREL**. (General Refine Expression Language). For the following step, review the OpenRefine manual for GREL (optionally also the Illinois reference).

9.1 Although the **To date** transformation converts values into the ISO compliant YYYY-MM-DD form, it also contains time information that we now want to remove. To do so, you decide to apply some custom regular expressions.

- Tasks:
  - **Add** a new column based on the **last_review** column first. Enter **last_review_timeless** for the new column name.
  - On the **last_review_timeless** column execute the following **operations:**
    - **Edit cells** → **Transform** → **toString(toDate(value),"yyyy-MM-dd")**
  - Now you have an ISO standard date format but without the time information!

9.2 For the **name** column,

- Tasks:
  - Add a new column based on the **name** column and name the new column as **name_grel**.
  - Using GREL (or by other means), remove **the outermost parentheses** in each name (but not the inner ones), i.e., where the first character is "(" and the last character is ")". For example, the desired outcome looks like this:

```
Original: (Lincoln Park (Oasis) - Unit 2 ONLY)
Cleaned: Lincoln Park (Oasis) - Unit 2 ONLY
```

Hint: search on OpenRefine recipes. https://github.com/OpenRefine/OpenRefine/wiki/Recipes
You might also want to refer back to the regular expression notes on how to express the beginning and ending anchors. Also note that to use GREL, you might have to add outermost slashes in order to effectively transform using regex (e.g. / abc+ /)

  - Now treat **the exclamation marks** (!) and **the asterisks** (*) as follows:
    - **Create** a new column based on the **name_grel** column and name it as **name_grel_star**
    - Use GREL to remove all exclamation marks and all asterisks.
    - Your desired outcome should look like this:

```
Original: *** Luxury in Chicago!!! 2BR/ 2Ba / Parking / *BBQ**!!
Cleaned: Luxury in Chicago 2BR/ 2Ba / Parking / BBQ
```

  - Close the text facet for the **name_grel** column after these tasks.

10. **ADVANCED FACETS**. Now that you've experienced the power of text facets, explore the use of numeric facets on your datasets.
    - Task 1:
        - Create a **numeric facet** for the **price** column
        - You noticed there are a lot of unreasonable prices for a listing. You want to inspect those that are **$5000 and above** per night. To take note of these outrageous listings, you can do this:
        - Adjust the range of the numeric facets to $5000 and up. You will see the table is being filtered according to the records in this range. Based on the **price** column, add a new column **price_crazy**, and in the expression box, enter "**1**".
        - Remove the numeric facet for **price** after this task.
        - You should notice that only the listings that are $5000 and above have been marked with '1' in the price_crazy column.

    - Task 2:
        - Create a numeric facet for the **minimum_nights** column.
        - Adjust the range of the numeric facets to "300 nights and above". You will see the table is being filtered according to the records in this range. Based on the **minimum_nights** column, add a new column **minimum_nights_long**, and in the expression box, enter "**1**".
        - Remove the numeric facet for minimum_nights after this task.

11. **TRIM the leading and trailing whitespaces**, as well as **COLLAPSING consecutive whitespaces** for the columns that are of type text one more time. These are: **name_grel, name_grel_star, host_name 1, host_name 2, neighbourhood_case, neighbourhood_loop,** and **neighbourhood_cluster**.

There are still a lot of "messy" cells in this dataset (e.g., non-standard characters in the name column), but you are happy with the state of affairs compared to the original dataset. You've completed the tasks, and now it's time to save your projects and move on with life ;-)
To push to the finish line, please complete Step 4.

STEP 4: Please refer to the Submission & Autograding guide for instructions on how, what, and where to submit the assignment files.

Congratulations! Hope you have a nice stay with your family in Chicago!