Data pre-processing documentation – Mohammed Aminoor Rhaman

Data pre-processing of police data

Pre-processing of all the police station data can be found on the 'Preprocessing_Police_Data.ipynb' file (will refer to as the *final file*).

On the file titled 'Preprocessing_test.ipynb' (will refer to as the *test file*), data cleaning was initially applied and tested on the file '2023-10-bedfordshire-street.csv'. After it was verified that the data cleaning was completed, the data cleaning steps was then applied to the entire dataset.

Below reads the steps that were carried out to clean and join all the data together.

0. The first pre-step was to import all the necessary libraries: os, glob, panda and numpy.

1. The os library was used to select the correct directory.

2. We first start by joining all the months for each police department. Starting with Hertfordshire, we use 'glob.glob' to search for each file ending in "*hertfordshire-street.csv". '**' lets us search through every subfolder.

3. After the Hertfordshire data has been concatenated, we preserve an original copy of the combined police data in case an error is made down the line during pre-processing.

4. Possible different null values ('?', 'NULL', 'NA', ' ') are standardized to the standard NaN (Not a Number) value that pandas uses to describe null values.

5. Rows which had 'No location' on the location column were removed using the 'query' method when handling datasets. These rows were removed as there is no way of knowing if the crime happened outside of the region covered by the police force. Carrying out this step removed most null values.

6. The context column was deleted using 'drop' as all values were null. This was verified using '.isnull().sum()'.

7. The ‘Falls within’ column was deleted using ‘drop’ as it was the exact same as the ‘Reported by’ column. This was verified using ‘value_counts’ in the test file.

8. The datatype of all the objects in the ‘Month’ column was changed to datetime datatype, using ‘to_datetime’ method. The result of this operation meant that each crime defaulted to the first day of the month (very important to keep in mind when plotting time on the x-axis for data visualisation).

9. Null data in the ‘crime ID’ column was filled with ‘NoCrimeID’ using ‘fillna’.

10. Null data in the ‘Last outcome category’ was filled with ‘not available’.

11. The final step involved filtering out rows which had ‘LSOA names’ that resided *outside* of the region covered by Hertfordshire police force. The reason such data exists is likely due to crimes which took place in another area, but the case was then transferred to in this case, the Hertfordshire police force due to the suspect or victim living within Hertfordshire. It’s unlikely to be entered incorrectly as it would imply that the latitude, longitude and LSOA code were also entered incorrectly. Using ‘.str.contains’, rows which did not include the names of the districts of Hertfordshire *(see at the end of the page for a list of all districts)* were removed.

12. Using ‘.value_counts’ and ‘.nunique’, we check that there only fourteen unique crime types (I.e. making sure there is no typos in this column).

13. Steps 2 to 12 are carried out for the Bedfordshire, Surrey and Kent data.

14. All the police force data is then combined via ‘.concat’.

15. A district column was added to the police force data. This was done so that during the EDA, data could be grouped by district. With the current dataset, the ‘LSOA name’ characterizes the zone *within* the district for each crime reported. For example, on the ‘LSOA name’ column, we may find ‘Bedford 001A’. What we need is just ‘Bedford’ from the string. To do this, ‘str.contains’ was used. This us search to see if ‘Bedford’ in this case is in the string: ‘Bedford 001A’.

A list of all the districts for each of the police forces:

Hertfordshire districts: Broxbourne, Dacorum, East Hertfordshire, Hertsmere, North Hertfordshire, St Albans, Stevenage, Three Rivers, Watford, and Welwyn Hatfield.

Surrey districts: Elmbridge, Epsom and Ewell, Guildford, Mole Valley, Reigate and Banstead, Runnymede, Spelthorne, Surrey Heath, Tandridge, Waverley, and Woking.

Bedfordshire districts: Bedford, Central Bedfordshire, and Luton.

Kent districts: Bedford, Central Bedfordshire, and Luton.

Note: Each district has many different LSOA codes which indicate different areas of the district (e.g. Broxbourne 001D). The filter used in step 11 checks to see if at least one of the correct district names is listed.

Data preprocessing of GDHI data

Before the GDHI data can be used for the analysis, adjustments have to be made to make data merges possible. The pre-processing file is titled: 'Preprocessing_GDHI.ipyynb. Below is a list of steps taken to filter and process this data:

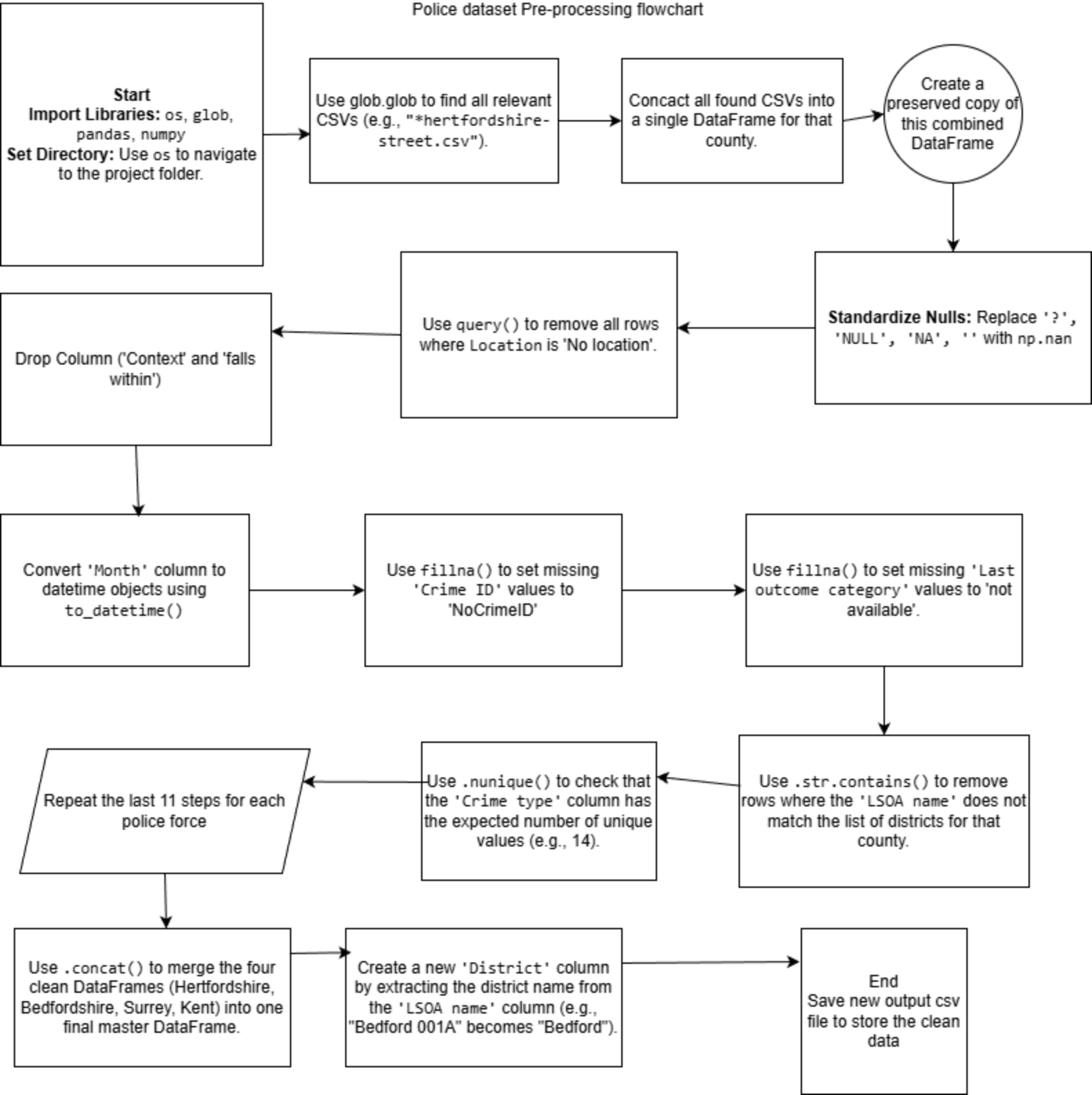
1. When opening the GDHI data, data in separate sheets were opened and stored as one sheet contained the GDHI per head whilst another sheet contained the population numbers.

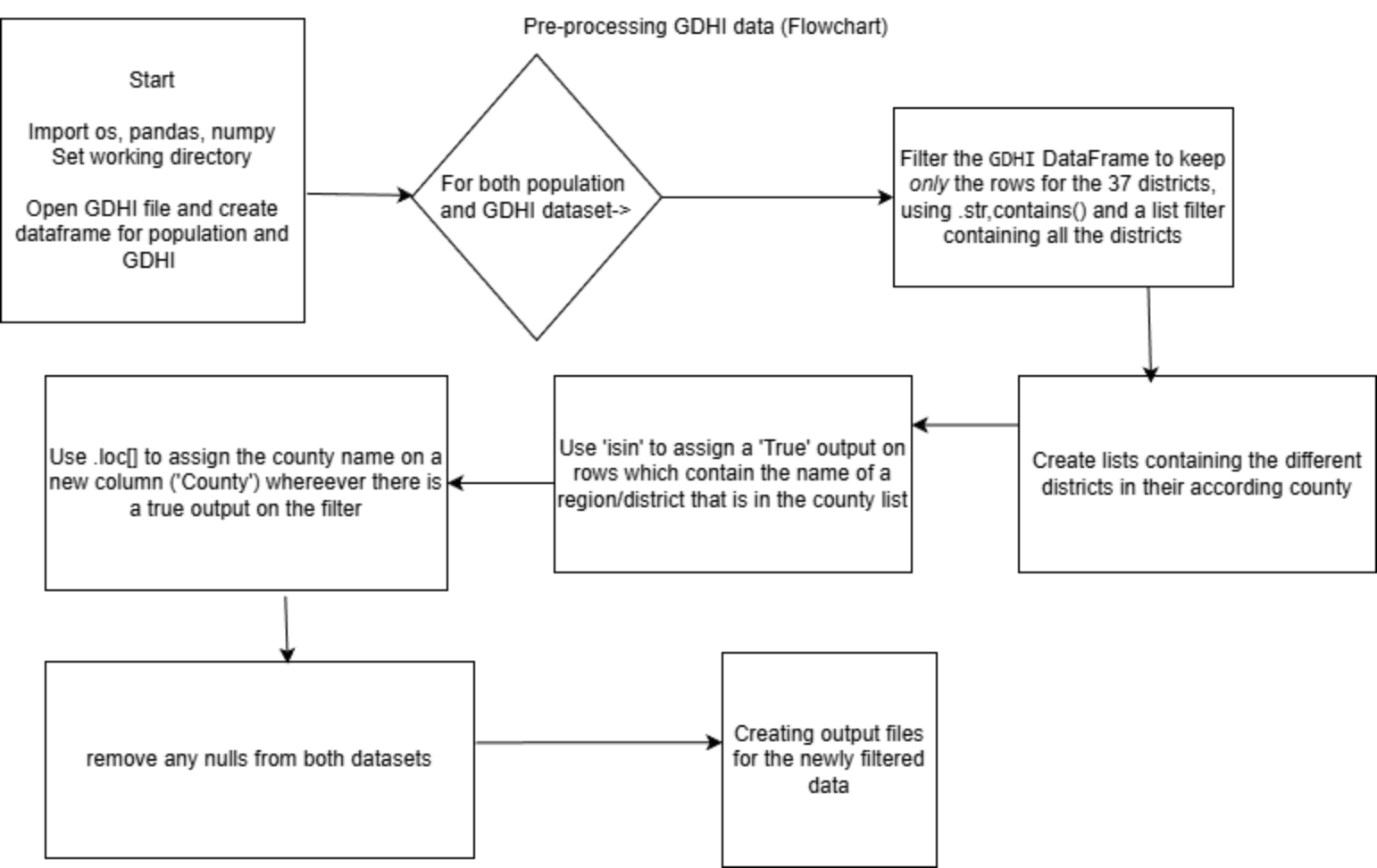
2. First step was to filter the rows for districts that are included in the counties we are interested in. This was done using 'str.contains'.

3. The last step was to then add a new column with the name of the county each row belonged to. This was done by creating lists that contain the districts in each of their counties and then applying the filter one by one using 'isin', which produces a Boolean output. '.loc' is then used to assign the county based off the output of the Boolean filter.

4. New output files were created in xlsx format.

Police dataset Pre-processing flowchart




Pre-processing GDHI data (Flowchart)
Start
Import os, pandas, numpy
Set working directory
Open GDHI file and create dataframe for population and GDHI
For both population and GDHI dataset->
Filter the GDHI DataFrame to keep only the rows for the 37 districts, using .str,contains() and a list filter containing all the districts
Create lists containing the different districts in their according county
Use 'isin' to assign a 'True' output on rows which contain the name of a region/district that is in the county list
Use .loc[] to assign the county name on a new column ('County') whereever there is a true output on the filter
remove any nulls from both datasets
Creating output files for the newly filtered data