Capstone Project 1: Data Wrangling

- After importing both data frames and looking of their structures, I noticed in particular one column of each data frame ('healthcares' and 'municipalities') that had a lot of null data (>100,000 and > 35,000 respectively). I decided to remove it (the values were not necessary for the project).

- The 'zip code' and 'insee code' columns, common in both data frames are very important because they could be useful to merge them later. Then, I decided to remove all the null values in both 'zip code' & 'insee code' columns in the two data frames.

- The France is composed of a continental territory and five overseas departments. The data frames include data for all these territories. In this study, we will focus only on metropolitan France. Thus, I deleted the data concerning the five overseas departments in both data frames.

- I noticed that the type of 'zip code' column of 'municipalities' dataset is still object despite the elimination of the null cells. To understand why, I sorted the data frame by the 'zip code' column in descending order. And I found ten rows that not contain a zip code but a string value. I deleted these ten rows.

- I noticed that some name of region in 'municipalities' dataset are false so I corrected them.

- I created 2 new columns that could be useful later in 'municipalities' dataset.
  - The first one is composed of the object values: 'north' and 'south' according to the location of the regions. There are 5 regions in the south and 8 regions in the north of the France.
  - The second one is composed of ten object values related to ten different groups of population number. I used the function .quantile to cut the population of each municipalities in 10 equal groups.

- Then, the important step here was to merge both data frames. We have two common columns in each dataset. The 'zip code' column or the 'insee code' column, another code for each municipality. The problem with the 'zip code' column is that each zip code could match with several different municipality and after the merge with a left join, we obtain too many rows. And the problem with the 'insee code' column is that we don't have all the data in the 'municipalities' dataset. After the merge, we obtain a lot of rows with null values in columns of 'municipalities' dataset. Finally, I used a merge with a left join to keep all the data in the 'healthcares' data frame, using both 'zip code' and 'insee code' columns. The 'healthcares' dataset has 149,477 rows and the dataset after the merge has 159,753 rows (and 146,224 rows with no null values for the columns belong to

‘municipalities’ dataset). But in this way, I combined both problems. This seems to not be the better solution.

- Another way to perform the merge is to use only the first three digits of each zip code in each dataset. I created a new column in each data frame with the first three digits of zip code. Then I created new datasets using .groupby, one counting the number of doctors for each unique value of the first three digits of zip codes and one with the mean of the population for each unique value of the first three digits of zip codes. Then, I merged the two new datasets using .join.