Rui Hu 1002147376
INF1340H: Programming for Data Science
Professor Shion Guha
Midterm Project – Data Cleaning Writeup
Nov 14, 2022

Introduction

Data cleansing is a key component of data analysis and modeling. It can be handled by using a variety of tools, such as Python, Excel, Tableau, etc. Inconsistencies can arise when transferring data from a well-formatted excel spreadsheet into Python. Restructuring the data frame and controlling the data type of variables allows data users to work with a well-organized data frame in further analysis. Additionally, controlling the data type of variables prevents errors in data entry and allows for more accurate data analysis.

In this project, I managed to reorganize the provided excel spreadsheet in python through multiple steps and to align the final output with five tidy data principles. In general, most tables require similar operations:

1. Stacking dataframe: transform multi-column variables to single-column variables
2. Handling missing values: replace with NaN
3. Splitting dataset: into multiple small datasets with logical meanings
4. Formatting output and unifying column data type

Details of each steps will be discussed in the following paragraphs.

Methods and Results

Appendix Table

The three appendix tables (CONTENTS, ANNEX, NOTES) are simple. These tables only require direct import from excel spreadsheet. One thing to notice is that original data contains a formatted title page for all tabs, ie. row 1 to row 15. I use “skiprows” under pd.read_excel to exclude these rows from the importing process. The reason for exclusion of these rows is that they are simply text and are not considered as a dataset.

	TABLE	TITLE
0	Table 1	International migrant stock at mid-year by sex...
1	Table 2	Total population at mid-year by sex and by maj...
2	Table 3	International migrant stock as a percentage of...
3	Table 4	Female migrants as a percentage of the interna...
4	Table 5	Annual rate of change of the migrant stock by ...
5	Table 6	Estimated refugee stock at mid-year by major a...
6	ANNEX	Classification of countries and areas by major...
7	NOTES	NOTES

	Country code	Country or area	Sort order	Major area	Code	Sort order.1	Region	Code.1	Sort order.2	Developed region	Least developed country	Sub-Saharan Africa
0	4	Afghanistan	99	Asia	935	71	Southern Asia	5501	98	No	Yes	No
1	8	Albania	154	Europe	908	127	Southern Europe	925	153	Yes	No	No
2	12	Algeria	40	Africa	903	7	Northern Africa	912	39	No	No	No
3	16	American Samoa	257	Oceania	909	238	Polynesia	957	256	No	No	No
4	20	Andorra	155	Europe	908	127	Southern Europe	925	153	Yes	No	No
5	24	Angola	30	Africa	903	7	Middle Africa	911	29	No	Yes	Yes
6	660	Anguilla	182	Latin America and the Caribbean	904	180	Caribbean	915	181	No	No	No

	FootNotes	NOTES
0	(a)	The column labeled “Type of data” indicates wh...
1	(b)	More developed regions comprise Europe, Northe...
2	(c)	Less developed regions comprise all regions of...
3	(d)	The least developed countries, as defined by t...
4	(e)	Sub-Saharan Africa refers to all of Africa exc...
5	(1)	Including Agalega, Rodrigues and Saint Brandon.
6	(2)	Including Zanzibar.
7	(3)	The estimates for 1990 to 2005 refer to Sudan ...
8	(4)	Including Ascension and Tristan da Cunha.
9	(5)	For statistical purposes, the data for China d...

Table 1-3, 5

Table 1, 2, 3, 5 have similar structures and I write one function melt_df to handle these tables. In the importing stage, I use “skiprows” to exclude the title page and use header = [1,2] to set multi-index column names. It will be easier in the later unstacking process.

```
df_1_ori = pd.read_excel('UN_MigrantStockTotal_2015.xlsx',sheet_name = 'Table 1', skiprows = 13, header=[1,2])
df_2_ori = pd.read_excel('UN_MigrantStockTotal_2015.xlsx',sheet_name = 'Table 2', skiprows = 13, header=[1,2])
```

Use Table 1 as an example, both year information and original column names are set as headers. I noticed that “International migrant stock at mid-year” is splitted into 3 large sections: Both sexes, male, and female. Therefore, I performed a multi-level melt to group value variables into “Gender” and “Year”.

	Sort\norder	Major area, region, country or area of destination	Notes	Country code	Type of data (a)	International migrant stock at mid-year (both sexes)					...	International migrant stock at mid-year (male)			
	Unnamed: 0_level_1	Unnamed: 1_level_1	Unnamed: 2_level_1	Unnamed: 3_level_1	Unnamed: 4_level_1	1990	1995	2000	2005	2010	...	2000	2005	2010	2015
0	1	WORLD	NaN	900	NaN	152563212	160801752	172703309	191269100	221714243	...	87884839	97866674	114613714	126115435
1	2	Developed regions	(b)	901	NaN	82378628	92306854	103375363	117181109	132560325	...	50536796	57217777	64081077	67618619

By looping through columns, I grouped columns into id_vars and value_vars. Id_vars are general variables and will not be changed in the “pd.melt” process. Value_vars are used to stack values into one single column.

```
id_vars
```

```
[('Sort\norder', 'Unnamed: 0_level_1'),
 ('Major area, region, country or area of destination', 'Unnamed: 1_level_1'),
 ('Notes', 'Unnamed: 2_level_1'),
 ('Country code', 'Unnamed: 3_level_1'),
 ('Type of data (a)', 'Unnamed: 4_level_1')]
```

```
value_vars
```

```
[('International migrant stock at mid-year (both sexes)', 1990),
 ('International migrant stock at mid-year (both sexes)', 1995),
 ('International migrant stock at mid-year (both sexes)', 2000),
 ('International migrant stock at mid-year (both sexes)', 2005),
 ('International migrant stock at mid-year (both sexes)', 2010),
 ('International migrant stock at mid-year (both sexes)', 2015),
 ('International migrant stock at mid-year (male)', 1990),
 ('International migrant stock at mid-year (male)', 1995),
 ('International migrant stock at mid-year (male)', 2000),
 ('International migrant stock at mid-year (male)', 2005),
 ('International migrant stock at mid-year (male)', 2010),
 ('International migrant stock at mid-year (male)', 2015),
 ('International migrant stock at mid-year (female)', 1990),
 ('International migrant stock at mid-year (female)', 1995),
 ('International migrant stock at mid-year (female)', 2000),
 ('International migrant stock at mid-year (female)', 2005),
 ('International migrant stock at mid-year (female)', 2010),
 ('International migrant stock at mid-year (female)', 2015)]
```

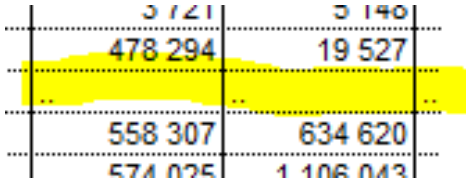
After “pd.melt”, I changed year and gender information from column names to 2 individual columns.

	(Major area, region, country or area of destination, Unnamed: 1_level_1)	(Notes, Unnamed: 2_level_1)	(Country code, Unnamed: 3_level_1)	(Type of data (a), Unnamed: 4_level_1)	Gender	Year	International migrant stock at mid-year
0	WORLD	NaN	900	NaN	International migrant stock at mid-year (both ...	1990	152563212
1	Developed regions	(b)	901	NaN	International migrant stock at mid-year (both ...	1990	82378628
2	Developing regions	(c)	902	NaN	International migrant stock at mid-year (both ...	1990	70184584
3	Least developed countries	(d)	941	NaN	International migrant stock at mid-year (both ...	1990	11075966

Next, I changed the column names to be more meaningful and removed “Unnamed”.

	Major area, region, country or area of destination	Notes	Country code	Type of data (a)	Gender	Year	International migrant stock at mid-year
0	WORLD	NaN	900	NaN	International migrant stock at mid-year (both ...	1990	152563212
1	Developed regions	(b)	901	NaN	International migrant stock at mid-year (both ...	1990	82378628

Going through the original data, I noticed some cells contain “..” to indicate missing value. However, when importing the dataset into python, “..” will be considered as string and other numbers will be considered as float. Thus the data type within one column is not consistent. I replaced all “..” with np.nan (read as float type and indicate missing value).



Lastly, “Gender” column contains too much information. By searching “both sexes”, “male”, “female” as indicators, i replace the gender column with “Both”, “M”, “F” to make sure each column only contains one variable.

	Major area, region, country or area of destination	Notes	Country code	Type of data (a)	Gender	Year	International migrant stock at mid-year
0	WORLD	NaN	900	NaN	Both	1990	152563212.0
1	Developed regions	(b)	901	NaN	Both	1990	82378628.0

Up to this step, I have changed the original dataset to align with most of the tidy data principles. Each column name is informative and doesn’t contain values. Each column contains only one variable and has a singular data type. All variables are in cells.

Table 4

For Table 4, since original data doesn’t have gender classification, thus I only stacked data based on year information.

	Major area, region, country or area of destination	Notes	Country code	Type of data (a)	Year	Female migrants as a percentage of the international migrant stock
0	WORLD	NaN	900	NaN	1990	49.039150
1	Developed regions	(b)	901	NaN	1990	51.123977

Table 6

For Table 6, there are three types of value variables and no gender classification. The “year” information also has two formats. It is important to split the two “year” type into two tables, otherwise the year column will have different data type.

Estimated refugee stock at mid-year (both sexes)					...	Refugees as a percentage of the international migrant stock					Annual rate of change of the refugee stock				
1990	1995	2000	2005	2010	...	1995	2000	2005	2010	2015	1990-1995	1995-2000	2000-2005	2005-2010	2010-2015
18836571	17853840	15827803	13276733	15370755	...	11.103013	9.164736	6.941389	6.932687	8.033424	-2.123497	-3.837069	-5.557223	-0.025089	2.947267
2014564	3609670	2997256	2361229	2046917	...	3.910511	2.899391	2.015025	1.544140	1.391085	9.388424	-5.983348	-7.277379	-5.323293	-2.087656

I created one specific function melt6_df for Table 6 and it split the dataset into two individual datasets based on “Year”.

Major area, region, country or area of destination	Notes	Country code	Type of data (a)	Year	Estimated refugee stock at mid-year (both sexes)	Refugees as a percentage of the international migrant stock
Afghanistan	NaN	4	B	1990	25.0	0.043338
Afghanistan	NaN	4	B	1995	19605.0	27.411146

Major area, region, country or area of destination	Notes	Country code	Type of data (a)	Year	Annual rate of change of the refugee stock
Afghanistan	NaN	4	B	1990-1995	128.993470
Afghanistan	NaN	4	B	1995-2000	NaN
Afghanistan	NaN	4	B	2000-2005	NaN
Afghanistan	NaN	4	B	2005-2010	102.911692

Country & Major Area split

The original data use color code and indention to separate major area and country data. This information disappeared when imported into python. To reduce the information loss through the data cleansing process, I decided to split each table into “Country” and “Major Area”. The main idea for this split is to check whether the “major area, region, country or area” column in each table exists in “Country or area” from “ANNEX”. If it does, then that row will be classified as a country and vice versa, as a major area.

After the split, the original column name is not informative anymore. I changed the column name to “Country” or “Major Area” correspondingly.

The value variables for all tables are either number of count or rate of change (%). Therefore, I formatted the final output for better visual experience. The submitted workbook will contain all tables and please check for more details.

Table 1: International migrant stock at mid-year by sex and by major area, region, country or area, 1990-2015

```
81] format_df(df_1_c,"N").head(10)
```

	Country	Notes	Country code	Type of data (a)	Gender	Year	International migrant stock at mid-year
8	Burundi	NaN	108	B R	Both	1990	333,110
9	Comoros	NaN	174	B	Both	1990	14,079
10	Djibouti	NaN	262	B R	Both	1990	122,221
11	Eritrea	NaN	232	I	Both	1990	11,848

```
] format_df(df_5_a,"P").head(10)
```

	Major Area	Notes	Country code	Type of data (a)	Gender	Year	Annual rate of change of the migrant stock
0	WORLD	NaN	900	NaN	Both	1990-1995	1.05%
1	Developed regions	(b)	901	NaN	Both	1990-1995	2.28%
2	Developing regions	(c)	902	NaN	Both	1990-1995	-0.49%

Discussion

This project helped to deepen my understanding of tidy data principles and to practice how to use python to accomplish basic data cleansing. I need to review the excel sheets multiple times to plan what steps are needed to create the desired outcome, what that outcome should look like, and how many manual steps are involved. I also need to identify any potential risks or problems that could occur during the process. The process of thinking and planning is more valuable than the final mark on this project. By taking the time to think about my long-term goals, I can make better decisions and plans. This process also allows me to develop relationships and connections that will be valuable down the road, and it allows me to learn new things.

I did a lot of syntax searches on google to write efficient and clean code. Although the primary goal of this project is to cleanse the data, I have been thinking about ways to make further analysis easier in the future. Documenting the data cleansing process can ensure that

the data is clean and that any step can be repeated if necessary. After changing the column names and moving all variables into cells, users can read the dataset without further clarification. A singular data type column dataframe is easier for row or column operation. For example, if the column consists of a string and an integer, it is impossible to find the sum of the whole column.

In general, I am satisfied with what I have done on the project but there are still several improvements that can be done additionally. For example, the dataset can be further divided into the country, regions, major areas, and others. Gender “Both” can be separated from “M” and “F” and into an individual dataframe. Table 6 can be split into three separate tables based on value variables instead of two. Add-in one more step to format the columns of each table, and manually set the data type for each column. By making all of these improvements, the final results will more closely match the five principles.

Conclusion

Achieving a satisfactory result in data preparation requires clear thinking and design. Data cleansing is an important step in that process. It is important to identify and correct errors in the data, to ensure that the data is complete and accurate. Data cleansing can be a time-consuming and difficult process, but it is essential to ensure that the data is of high quality.

Following the tidy data principle is a good start, but there are other things to think about such as missing values, duplicates, and choosing which data to use. In reality, data can be way more complicated and messy than project data. Data can be unstructured, with many different formats and sources. It can be difficult to clean and normalize. But with the right tools and techniques, it is possible to make sense of even the most complex data sets. Finding the pattern within a large chunk of data and organizing the dataset into input for the next step are crucial skills. I learned a lot from this project and am willing to use what I have learned in my future career.