Gabriel Marken

Cleaning the data. The first thing I did was to check which columns have missing data. This won’t tell us about erroneous data, but it’s a start. I also checked to see what the dtypes each column should hold. This will be useful later when I clean the data. Next I created a bool series which I figured out how to do online. This allowed me to check all the null data in a given column. I checked each of the four columns which might have had missing or incorrect data. This confirmed what I found above that only ‘Order Priority’ and ‘Order Type’ had missing data. It also told me which rows had the null values. Next I created a clean data variable where I dropped three columns on the end which had no data at all. I then changed all the missing data to the string ‘NULL’. After that I dropped all rows which contained the string ‘NULL’ which left me with 49979 rows.

Now that we have gotten rid of all the missing data I started checking for all rows which had non numeric data in ‘Order ID’. I found five rows. I moved on to the country column and checked if the strings contained any numbers by searching if it contained at least a single digit of a number, 0-9. I found an additional three rows. I changed all the incorrect ‘Country’ values to ‘NULL’ and all the incorrect ‘Order ID’ values to ‘0’ and deleted any rows which contained a value of either. This left me with 49971 rows and a clean dataset. Finally, I wrote this data to a csv called MM\_Sales\_clean.csv

In part 2 it was time to find some information about the data. I called the clean data and saved it as newData. I had to look at data from two columns so I grouped them using groupby. I found how to do this on stack overflow. I converted it to a dataframe since I had a better understanding of how to manipulate them. I was able to find the top 10 countries by units sold using .head(10) of ‘Units Sold’ and sorting by ascending=false. In number 1A, I took the info I found above and used seaborn to visualize the data in a bar graph similar to what we have done in class. In number 1B I opened/appened the info found above to a file called MM\_Rankings.txt. To get it to look how it was supposed to, I added a new top line and then deleted what was now the second line. This was formerly the names of the columns. Next comes a lot of really chunky code which is simply meant to clean up the text file. I open the file, search for a particular section of code within the txt file, and write over it with a cleaner bit of text. The txt file would look awful without this section. Finally, I print a new line and append a recommendation of what to do, given the data found.

In number 2A, I created a new variable which got a count of the ‘Sales Channel’ column. In 2B I did the exact same thing as 2A but for the ‘Order Priority’ column. Next, in 2C I created a pie chart in matplotlib which showed the distribution of the different order priority use counts. In 2D I added the info from sChannel (sales channel count) and op (order priority count) to the MM\_Rankings.txt file in the same way I appended info earlier. This time however, I didn’t have to deal with all that messy code.

In number 3A, I used seaborn to create a box plot of ‘Item Types’ and ‘Total Profit’. It was nearly identical to an example we had done in class so I didn’t have to do much googling for this one. 3B asked for the sum of profit by item type. I used the groupby function like I did earlier to connect the two values and printed the results. 3C wanted a visualization of item types by occurance so I used a bar graph method that I found online which would allow me to count the number of times each item type appeared in the file then show it in a graph. 3D asked for the top 3 most profitable item types per unit. For this I had to find the total profit of each item type brought in, then divide by the number of items used to bring in that profit. Next I turned the results into a df and printed them in descending order and only printed the top three. Next I turned it into a list. I honestly don’t remember why, I think I thought it would be easier to add it to the txt file. 3E asked me to append this info to the MM\_Rankings.txt file, so I used the same method as before to do this. I was also left with having to use the bulky method I used above to clean up the txt file and make it more readable. Again, I searched for particular strings which had white space and replaced them with cleaner versions of themselves. 3F asked me to discuss my findings, so I did.

In number 4 I was tasked with finding some descriptive statistics about the data. First in 4A I summed the data from the columns asked for and printed the results. 4B simply asked for the average of the same columns so I used the .mean to find the average of each column individually. 4C asked for the max value in each column so I used .max to find this. Next I turned the results from each of the three previous answers into dataframes so I could more easily work with the data. 4D asked me to create a line chart of the different sums. I put in each sum variable for each column and used matplotlib to visualize the sums of the different columns. It also asked for a second chart which had two lines, max and averages. In this I assigned averages to Y and max values to a z value. Now I had three values to work with which allowed me to put two lines on the same chart. 4E asked to put all the descriptive stats into a new txt file called MM\_Calc.txt, so I used the same method as before to append my results to the file once I had created it. Again, I was forced to use the bulky method of deleting white space and unwanted numbers to get the desired result.

Part 3 had two parts. Part one asked me to create a list of the regions and the countries in the region. It also asked me to return this as a dictionary with the keys being the regions and the list attached being the countries. First I used .uniques to find all the unique regions. Then I used groupby to connect regions to countries to find which countries were in each region. The result was unfortunately really hard to read so I converted it to a dataframe, then converted the dataframe to an extra text file which I could use to better read the data. Next, I created a dictionary and used the regions as keys and then input a large list of each country. I tried to create separate lists of each country and place them in here to clean up this part of the code, but the output kept giving me all the countries in a single cell which was problematic. After a few days of searching, I still couldn’t find a way to expand it, so I input each country individually which although time consuming, gave me the desired result in the end. 3B was the final section, and it asked me to take my results from 3A and write it to a csv file called Countries\_By\_Region.csv which I did, finishing the project.