Khanh Quach

Python Project - Marvel Mart Project

BUAN 4210 - 01: Programming and Data Management for Business Analytics

Documentation for project 2: Marvel Mart

Part 1: Cleaning the Data

In the beginning, I decided to use Pandas and Numpy to clean the data. After reading through the original file, I saw that there are some missing data at first glance, and based on the information about the **Country, Item Type, Order Priority, and Order ID.** The reason I know about this is by using the mm_sales.info() function to know about the characteristic of the data. This allows us to know how many items are missing in the CSV file. In the first step, I used the CSV and Pandas libraries to read the original file into a data frame, made a copy of the data frame to a new variable called mm_salesClean and then filtered out the error data using the function **mm_sales.isna().sum().** After that I create a copy of the current file so that I can make changes without impacting the original data. I clean all the data on the new file called **MM_Sales.csv.** After this, I know that there are **6 items** that have errors in the 'Item Type' and **15 items** that have errors in. After this, I decided to clean the 'Item Type' and 'Order Priority' first because, as the prompt said:

- Item Type (either missing AND/OR won't be a valid Item Type from the other ones listed)
- Order Priority (either missing AND/OR won't be a valid priority code of 'C', 'H', 'M', 'L', or 'NULL')

The fillna() method replaces the NULL values with a specified value. The fillna() method returns a new DataFrame object unless the in-place parameter is set to True. In that case, the fillna() method does the replacing in the original DataFrame instead. After that, I clean the ‘Country’ variable and the ‘Item Type’ using ‘try’ and ‘except.’

The try and except for Country value allow: It does this by attempting to convert each value in the 'Country' column to a float using a for loop to iterate through each row in the DataFrame. If the conversion is successful, it means that the value is not a string and is, therefore, erroneous. In this case, the value is replaced with the string "NULL," and the updated value is printed to verify the changes. I do the same with the ‘Order ID’ variable.

After all of our data is corrected. I decided to replace all missing data with 0 or “NULL.” After the all the values changed then, I decided to check once again that we got exactly the same number of data in the new cleaning file by using info() once again to make sure its all correct. Since the result file still had the column of number index, therefore I used index = false to delete the automatic index number that they created in the file. (Which allows keeping the shape of the data the same as the original file)

Part 2: Exploratory Data Analysis with Reports & Visualizations

For the second part of the assignment, I decided to use information that I learned in the last week of class to make a graph requirement model. **For question 1**, In order to find the top 10 countries with the highest sales, I decided to use function **n.largest(10,’all’) .** The 'nlargest()' method is then used to extract the top 10 countries with the highest number of sales from the numberofSales variable. These top 10 countries are stored in the 'top10Sales' variable, which is printed using the 'print()' function. The code I use after to create is the 'plt' module from the 'matplotlib' library to create a bar chart of the top 10 sales countries.

The next step is to write the results into MM_Ranking.txt. Using the same way as the one, I did in the above Part 1.

For question 2, in order to find the ‘Sales Channel,’ ‘Order Priority.’ I decided to use the count function built-in to find the number of Online and Offline here. In order to determine the count of online and offline use orders and the count of Order Priority types, the same method as in question 1 was used by applying. groupby() and .count(). sales channel grouped the Order IDs to determine the count of online and offline orders and by Order Priority types to determine the count of orders for each type of Order Priority. I created two data frames for these results to plot pie charts, and Matplotlib was used to plot the pie charts. The autopct was set to %.2f while plotting to display the two decimal places. The same appending method used previously was used to add information about the Sales Channels take-up and the Order Priority take-up to MM_Ranking. **For question 3,** To visualize the Profit Distribution by Item Types, I use a boxplot created using Seaborn. In this case, the y-axis is the scale of 1e9. It might make the data hard to see in Fruit. This happens due to the huge number of data collected, causing a big number of. **For question 4,** In order to find the sum, average, and maximum values for the Units Sold, Unit Cost, Total Revenue, Total Cost, and Total Profit, I used functions (sum), (mean), and (max). When printing each result, I use round() was added to round it to 3 decimal places. In order to find the sum and the averages, and the maximum, I relied on Matplotlib to do. A pandas data frame was generated for the Sum of each desired column (Total Revenue, Total Cost, and Total Profit since the requirement only these 3 variables to plot indicated it), as well as a data frame that had the Average and Maximum for those columns, before I plotted the two line graphs. I create and write the result to the file called MM_Cal.txt in order to write down the above analysis data.

Part 3: Cross-Reference Statistics

For the final part, I group the country by region using the Groupby() method to group countries altogether. I used unique() to make sure there won’t be any duplicates in Regions and Countries. This ensures that only unique region and country pairs are returned in the output. To make the resulting data more accessible for later use, I use a dictionary created from the resulting data using the 'to_dict()' method. The use of a dictionary allows me to store the unique region and country pairs as keys and values. To further analyze the data, I used the Pandas DataFrame with the dictionary to write it into the CSV file named ‘Countries_By_Region.csv.’ I use the index = false directive to ensure that the file will not automatically construct the index column. This directive will also prevent the creation of a new index for the column.