DATA STRUCTURES AND ALGORITHMS FINAL REPORT

Moksha Shah, Naisha Sheth and Shanaya Bajaj
30/04/2024

1. Introduction

The Dataset we chose is the Amazon Electronic Product Sale Data Set which was taken from kaggle. It contains 10,48,576 entries and 9 columns, of 19 years from 1999-2018. Below attached are the first few rows of the dataset to display the columns and the datatypes of it.

1	item_id	user_id	rating	timestamp	model_attr	category	brand	year	user_attr
2	0	0	5	13-06-1999	Female	Portable Audio & Video	Bose	1999	Female
3	0	1	5	14-06-1999	Female	Portable Audio & Video	Bose	1999	Female
4	0	2	3	17-06-1999	Female	Portable Audio & Video	Bose	1999	Female
5	0	3	1	01-07-1999	Female	Portable Audio & Video	Bose	1999	Female
6	0	4	2	06-07-1999	Female	Portable Audio & Video	Bose	1999	Female
7	0	5	2	12-07-1999	Female	Portable Audio & Video	Bose	1999	Female
8	0	6	5	13-07-1999	Female	Portable Audio & Video	Bose	1999	Female
9	0	7	2	13-07-1999	Female	Portable Audio & Video	Bose	1999	Female
10	0	8	4	16-07-1999	Female	Portable Audio & Video	Bose	1999	Female
11	0	9	5	20-08-1999	Female	Portable Audio & Video	Bose	1999	Female
12	0	10	1	24-08-1999	Female	Portable Audio & Video	Bose	1999	Female
13	0	11	1	04-10-1999	Female	Portable Audio & Video	Bose	1999	Female
14	0	12	5	10-10-1999	Female	Portable Audio & Video	Bose	1999	Female
15	0	13	5	12-10-1999	Female	Portable Audio & Video	Bose	1999	Female
16	1	14	4	17-10-1999	Female	Computers & Accessories	HP	2000	Female
17	0	15	5	21-10-1999	Female	Portable Audio & Video	Bose	1999	Female
18	2	16	4	25-10-1999	Female&Male	Headphones	Bose	2000	Female
19	0	17	5	29-10-1999	Female	Portable Audio & Video	Bose	1999	Female
20	0	18	1	06-11-1999	Female	Portable Audio & Video	Bose	1999	Female
21	0	19	3	09-11-1999	Female	Portable Audio & Video	Bose	1999	Female

Reasons for choosing this Dataset:

- Our curiosity towards consumer behavior with respect to electronic products. To see the difference in purchases from 1999 to 2018, through technological development.
- Potential in the variables provided and then trends we could analyze based on them.

2. Trends Analyzed

Based on the columns we analyzed 5 trends-

1. Product Performance Analysis
2. Brand Performance Analysis
3. Gender Preference Analysis
4. Yearly Analysis

5. Product Rating Analysis

3. Data Structures and Algorithms used

Vectors:
Dynamic Size: Vectors provide dynamic resizing, allowing for flexible storage of data without pre-determining the size. In scenarios where the amount of data is not known in advance, vectors prove to be advantageous
.
Sequential Access: Vectors offer efficient sequential access, enabling straightforward iteration over elements. This property is crucial for tasks like calculating yearly sales and average ratings, where data is accessed in a linear manner.
Random Access: Vectors support random access to elements using indices, facilitating quick retrieval of data at specific positions. This feature is beneficial for operations such as sorting and accessing elements by year.

Unordered Maps:
Fast Lookup: Unordered maps provide fast average-case lookup time, making them suitable for scenarios where data retrieval based on keys (e.g., brand names, years) is frequent. This property is crucial for tasks like counting sales by brand and identifying top-selling product categories.

Flexible Key-Value Storage: Unordered maps store key-value pairs, allowing for efficient storage and retrieval of associated data. This flexibility is essential for mapping relationships between data entities, such as mapping brands to their respective sales counts.

- *Bubble Sort:* It is one of the simplest sorting algorithms present. It works by swapping adjacent elements if they are in the wrong order. We used bubble sort because of its simplicity and the fact that it does not require additional memory space.

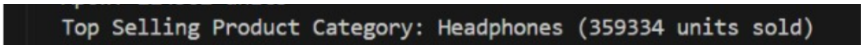
4. parseCSV Function

This function reads the dataset which is in the form of a CSV file. It parses through each line and extracts the data fields. We have used string stream to convert string representations to integers for an easier analysis. After parsing through the data, it is stored into the ‘SalesData’ struct and returned as a vector for further analysis.

5. Product Performance Analysis

This Trend uses the Product Category column to calculate the category with the most sold products over 19 years.

- This function uses an unordered map called ‘categorySales’ to store the data. Product category (string) and Count of Product category (integer) is stored into a key.
- It iterates through the sales_data vector and increments the count of the product in the ‘categorySales’ map. If the category is encountered for the first time it adds a new entry to the map with the count of 1.
- To find the category with the highest count we initialize ‘maxSales’ to store the maximum sales count and ‘topCategory’ that stores the name of the highest selling category.
- Next, it iterated over each key-value pair in the map. If the count of the current category (pair.second) is greater than the maximum sales count, it updates both the functions, ‘maxSales’ and ‘topCategory’ accordingly.
- After running the code we get the output as “Headphones” with 3,59,334 units sold.



6. Brand Performance .

- Data Processing: It uses an unordered map (brandSales) to store each brand and the count of sales for that brand.It iterates through the sales data, updating the count of sales for each brand in the map.

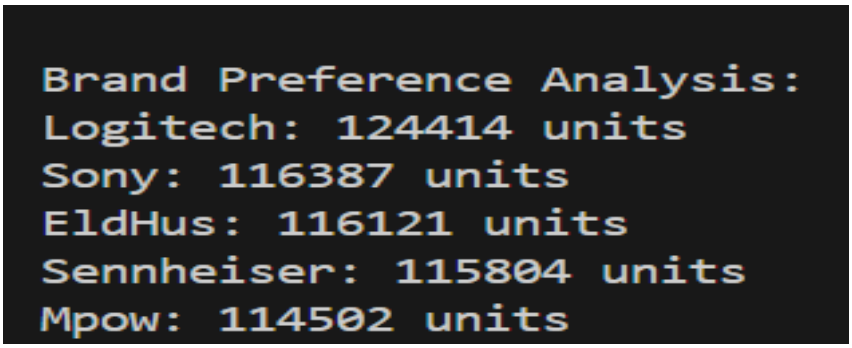
- Sorting: It converts the unordered_map into a vector of pairs (sortedBrands) to prepare for sorting.Then, it sorts the vector of pairs based on the sales counts using a custom sorting function (customSort)

- Output: It outputs the top 5 brands and their sales counts.It prints the brand names and their corresponding sales counts

- Data Structures Used : unordered_map<string, int>: Used to store brand names as keys and their corresponding sales counts as values. vector<pair<string, int>>: Used to hold the brand-sales count pairs for sorting

- Logic and Findings: The function aggregates sales data by brand to identify popular brands.Sorting the brands based on sales counts allows the function to identify the top 5 brands in terms of sales.

The analysis helps businesses understand customer preferences and tailor marketing strategies or inventory management accordingly. The function helps in identifying which brands are performing well in the market and which ones may need more attention or promotion.



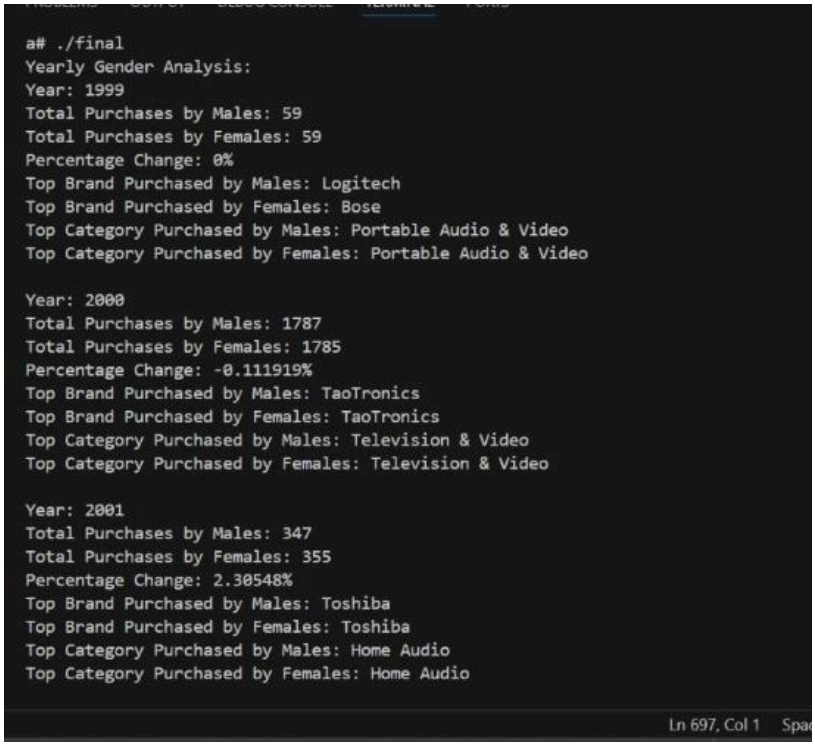
7. Gender Preference Analysis

In the code for gender analysis, we wanted to find the total number of sales by males that year , total number of purchases made by females and the % change is the difference between female and male purchases.
We then found the top brand that both genders bought from that particular year and the top selling product for them respectively.

- The gender function iterates through each row in the data vector.
- For each entry we wanted the function to check the user attribute to determine the gender of the user (male or female). We assumed,
  i) if the % change is 0 then it means there were equal purchases made by both.
  ii) if the percentage is positive it means females have made a greater purchase.
  iii) if the % is negative it means there has been a greater purchase made by males.

- Based on the gender, it updates the respective counters for total purchases by year and identifies the top brand for that gender in that year. We get the top brand by comparing the current entry's brand with the existing top brand for that gender and year. If the current entry's brand is greater, it becomes the new top brand.

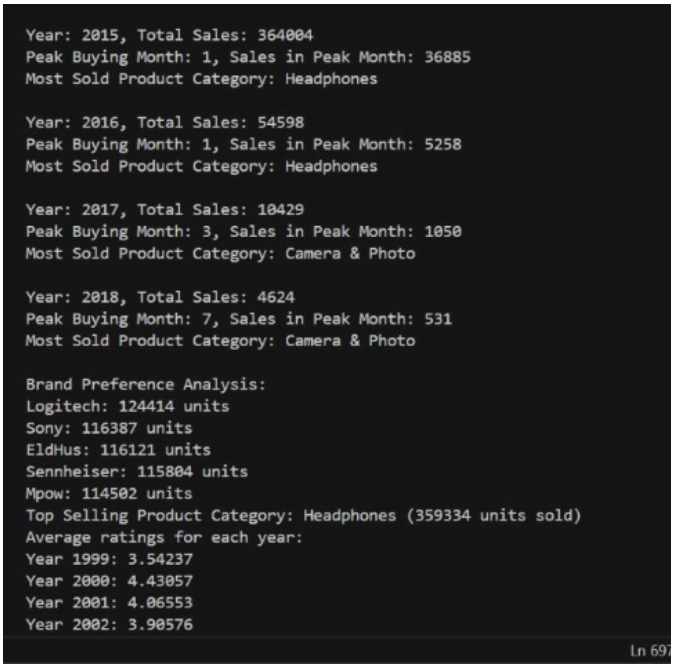
- Results:
  After processing all data rows, the function outputs the analysis results for each year.
  By printing the year and total purchases made by males and females for that year.
  Then calculates the percentage change in purchases between females and males for that year. Lastly it gives the top brand purchased by males and females for that year.



8. Yearly Analysis

In this function, we calculate yearly sales, find peak buying months, and identify the most sold product categories for each year.

- The calculateYearlySales Function, calculates the yearly sales by iterating through the sales data. It keeps a vector of pairs, in this each pair contains the year and total sales for that year. It examines each entry in the sales data to see if the year already exists in the vector. If it does, it increases the total sales for the year. If not, it creates a new entry for the year with a total sales count of one. After going through all of the data, it returns a vector containing yearly sales figures.
- findPeakMonth Function: This function finds the peak buying month and the sales in that month for a specified year. It initializes a vector to store sales for each month of the year. It iterates through the sales data and increments the sales count for each month. After counting sales for each month, it finds the month with the highest sales and returns the month number and the corresponding sales.
- findMostSoldProduct Function: This function finds the most sold product category in a specified year with the use of an unordered map to store the sales count for each product category. Then it iterates through the sales data, counting the sales for each category in the specified year. After counting sales for each category, it identifies the category with the highest sales count and returns it.



9. Product Rating Analysis

- Initialization: It initializes two vectors, avg_ratings and counts, both with a size of 2024 and all elements set to 0. These vectors will store cumulative ratings and the count of ratings for each year, respectively.

- Data Accumulation: It iterates through the sales data, accumulating ratings and counts for each year. For each entry in the sales data, it adds the rating to the corresponding year's cumulative rating in avg_ratings and increments the count for that year in counts.

- Calculation of Yearly Averages: After accumulating ratings and counts, it calculates the average rating for each year. It divides the cumulative rating for each year by the corresponding count of ratings. If the count for a particular year is 0 (indicating no ratings for that year), the average rating for that year remains 0.

- Output: Finally, it returns a vector yearly_avg_ratings containing the average ratings for each year.

10. Comparison with Other Data Structures

- Binary Search Trees: While binary search trees offer efficient lookup and insertion, they might not be as suitable for scenarios with frequent data insertion and deletion, as balancing operations can be costly. Vectors and unordered maps, being contiguous and hash-based data structures respectively, provide better performance for such tasks.

- Arrays: Arrays have fixed sizes and do not offer dynamic resizing, making them less suitable for scenarios where the amount of data varies. Vectors, with their dynamic resizing capability, provide a more flexible alternative.

- Stacks, Queues, Deques, Linked Lists: These data structures are optimized for specific operations (e.g., LIFO/FIFO access), which are not the primary focus of the analysis presented in the code. Vectors and unordered maps offer more versatility and efficiency for the diverse set of operations required in this scenario.

11. Limitations/inconsistencies

We noticed a drastic fall in the sales of amazon from 2016-2017 and larger fall from 2017 to 2018 which made it seem unrealistic when plotting our data in a graph for visual representation.This could have happened due to several factors , one being there was a shift or a mistake in the data set where all purchases made that year were not mentioned and also that the year 2018's data was not completely entered. This is the conclusion we came to after further analysis of our data set , as such significant shifts cannot realistically be caused just by shifts in the market at that time , it would have to be an issue related to the entries in the dataset.

12. Conclusion

The use of data structures and algorithms to analyze the Amazon Electronic Product Sale Dataset has given us valuable insights into consumer behavior and market trends over a span of 19 years, from 1999 to 2018. By employing various data structures such as vectors and unordered maps, alongside algorithms like bubble sort, we were able to efficiently process and analyze the vast amount of sales data contained within the dataset.

The trends analyzed covered a wide array of aspects including product performance, brand performance, gender preferences, yearly analysis, and product rating analysis.

Furthermore, the product rating analysis offered an understanding of consumer satisfaction trends over time, allowing businesses to gauge the performance of their products and make informed decisions regarding product development and marketing strategies.

While the chosen data structures and algorithms proved effective for the analysis conducted, there were limitations and inconsistencies encountered throughout the process. These included challenges in handling missing or incomplete data, as well as limitations in scalability when dealing with larger datasets.