

Table of Contents

[1. Problem Statement 2](#_Toc179047228)

[1.1 Initial Data Set 2](#_Toc179047229)

[2. Data cleansing 3](#_Toc179047230)

[2.1 Data cleansing: focus on data structure 3](#_Toc179047231)

[2.2 Data cleansing: Code interventions 5](#_Toc179047232)

[3. exploratory data analysis 6](#_Toc179047233)

[3.1 Customer Base 6](#_Toc179047234)

[3.2 Product Category Distribution 7](#_Toc179047235)

[3.3 Total Revenue per month 7](#_Toc179047236)

[4. Machine learning 8](#_Toc179047237)

[5. conclusion 9](#_Toc179047238)

# Problem Statement

The Numpy Supermarket chain, with locations in Windhoek and across Namibia, faces difficulties in leveraging its extensive customer transaction data for informed decision-making. Despite the availability of valuable data, a lack of effective data analysis is preventing the supermarket from optimizing service delivery, product offerings, and customer satisfaction. By applying data analytics techniques—such as data cleaning, preprocessing, and predictive modeling—the supermarket seeks to uncover actionable insights into customer behavior. This will enhance operational efficiency, boost customer loyalty, and improve overall business performance.

# Initial Data Set

The initial data set comprised of 16 columns made up of various data types. The table below provides the details of the columns as well as the descriptions of their respective data types. A deeper analysis of the initial dataset found that some of the data types were inappropriate for the purposes of our analysis. Moreover, keeping the initial dataset would have resulted in performance inefficiency and data handing challenges, thus a decision was made to cleanse the data.

**Table 1. Initial Columns and Data Types**



# Data cleansing

To ensure data quality and consistency, a data cleansing exercise was conducted on the initial columns and data types. A top-down approach was applied which focused on data structure first. By examining each column’s data type and its appropriateness for the data it holds we were able to ensure that data consistency was enforced at a higher level across the dataset. Furthermore, by assigning specific data types, we were able to impose constraints on what values are acceptable within each column. For example, converting the data type “object” to datetime64 for the **Date** column. Data type inconsistencies, such as dates stored as strings or numeric data as text are high-level issues and resolving these first simplifies later steps because we ensure that all data points in a column follow a single format. Section 2.1 below discusses in detail the steps taken in our top-down approach, while Table 2 provides a summary of our updated columns and data types. Section 2.2 briefly discusses further data cleaning outside of the focus on data structure. These interventions involved coding in Jupyter Notebook to further enhance our data cleansing efforts.

# Data cleansing: focus on data structure

1. **CustomerID (int32)**

* **Cleansing Applied**: Converted from the initial float64 to int32 to ensure IDs are whole numbers. Any missing or invalid IDs will be addressed, as they are crucial for identifying unique customers. Possible duplicates should also be checked and removed if necessary.

1. **Date (datetime64[ns])**

* **Cleansing Applied**: Converted from the initial object to datetime64 to facilitate date-based operations. Missing or improperly formatted dates should be corrected. Any date values that fall outside of expected ranges (e.g., future dates) should be flagged for review.

1. **Time (timedelta64[ns])**

* **Cleansing Applied**: Stored as timedelta64 to allow precise time calculations. Ensure that time data is uniformly formatted and check for any unusual time entries (e.g., negative durations or times that fall outside of business hours).

1. **StoreLocation (object)**

* **Potential Cleansing**: Standardize the text format (e.g., ensure all locations use consistent spelling and case) and check for any erroneous entries. Converting to category can also help identify unexpected values, as they will appear as outliers.

1. **ProductID (int32)**

* **Cleansing Applied**: Changed from the initial float64 to int32. This ensures IDs are whole numbers. Validate for unique IDs and cross-reference with the ProductName column to confirm accurate pairing of IDs and names.

1. **ProductName (object)**

* **Potential Cleansing**: Correct any spelling errors or inconsistencies in product names. We might also normalize the names for uniformity and remove any leading/trailing spaces.

1. **ProductCategory (object)**

* **Potential Cleansing**: Standardize category names to ensure consistency (e.g., "Produce" vs. "produce").

1. **Quantity (int32)**

* **Cleansing Applied**: Converted from the inital float64 to int32, assuming only whole quantities are allowed. Checked for negative or zero values where these are not expected, as they may indicate data entry errors.

1. **UnitPrice (object)**

* **Potential Cleansing**: Since this remained an object, consider converting to a numeric type (float64) for price calculations. Clean up any currency symbols or formatting issues and handle any missing or non-numeric entries.

1. **PaymentMethod (object)**

* **Potential Cleansing**: Standardize entries (e.g., "debit card" vs. "Debit Card").

1. **CustomerAge (float64)**

* **Cleansing Applied**: Ensure there are no negative or unrealistically high age values, as these may indicate errors. Missing values were be handled appropriately, possibly by filling with median or mean age.

1. **CustomerGender (object)**

* **Cleansing**: Standardize the values (e.g., "M" vs. "Male") and ensure only valid entries are present.

1. **CustomerFeedback (object)**

* **Cleansing**: Since this is free-form text, perform basic text cleansing like removing extra spaces and correcting obvious spelling errors.

1. **DiscountApplied (object)**

* **Potential Cleansing**: Standardize entries (e.g., 'Yes' vs. 'yes') and consider converting to a boolean type for easier logical operations. Handle any missing values, possibly by setting to False where appropriate.

1. **MembershipStatus (object)**

* **Cleansing**: Ensure consistent terminology (e.g., "Member" vs. "Yes").

1. **Country (object)**

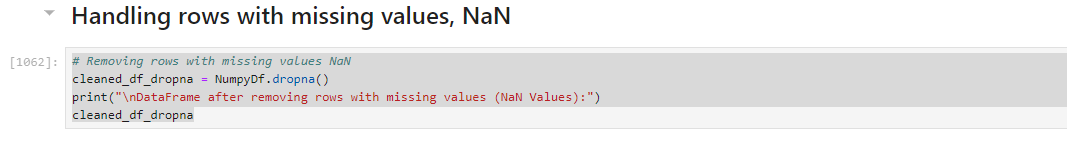
* **Cleansing**: Column was dropped as all branches currently only exist in Namibia.

**Table 2. Columns and Data Types post cleansing**



# Data cleansing: Code interventions

* + 1. Removal of Incomplete Data

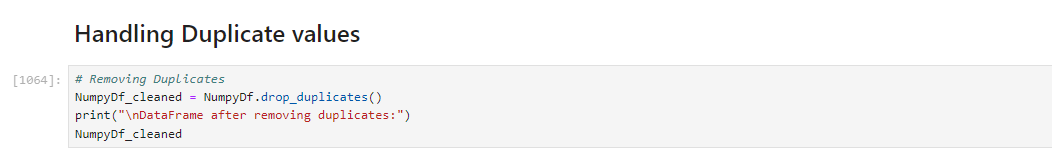


This code snippet was used to remove rows with missing values (NaN) from our DataFrame. The dropna() method is called on the DataFrame NumpyDf. This method is used to **remove rows or columns with missing values (NaN)**.

A new DataFrame called cleaned\_df\_dropna was then created to hold the results of NumpyDF.dropna(). The following print statement outputs a message indicating that the next output will show the DataFrame after NaN values were removed.

Cleaned\_df\_dropna displayes the new DataFRame which no longer contains any rows with missing values.

* + 1. Removing duplicates



This code snippet was used to remove duplicate rows from the DataFrame. The drop\_duplicates() method is called on the DataFrame NumpyDF to remove duplicate rows from the DataFrame. The results of after the duplicate rows have been removed are stored in NumpyDf\_cleaned.

The following print statement then outputs a message indicating that the upcoming output will show the DataFrame post duplicate rows having been removed.

NumpyDf\_cleaned shows the new DataFrame which no longer contains any duplicate rows.

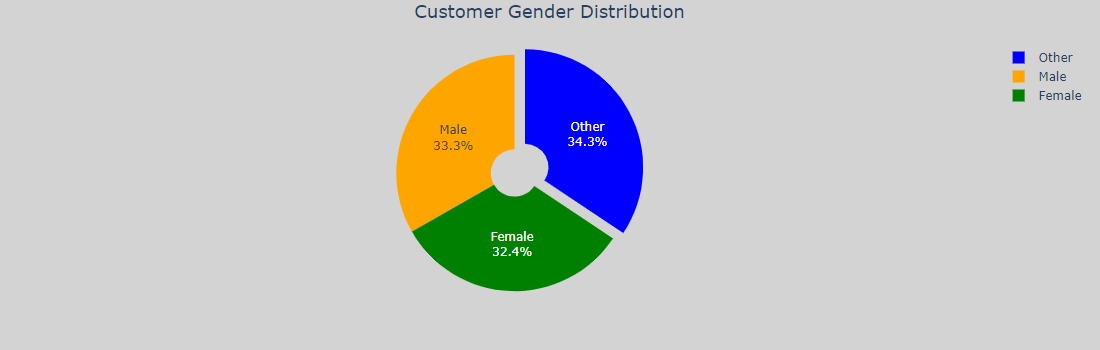
# exploratory data analysis

With the data finally clean we set out to perform exploratory data analysis that would help Numpy Supermarket answer questions about it’s customer base spread across all regions as well as their gender dynamics. Furthermore, Numpy Supermarket sought to find out the relative popularity across major product categories as well as the balance of distribution across the various regions which could help understand customer preferences and assist with inventory management. Finally, the Supermarket sought to understand its revenue performance relative to the various times of the year.

# Customer Base

The Numpy Supermarket customer bases comprise of a total of 2,300 clients that fall into 1 of the following categories: Female, Male or Other. The data in **Chart 1** below shows that the dominant customer is that of the other classification with 790 clients identifying with this category. The second largest type of customer identifies as male with a total of 765 clients, while females make up the smallest group of customers with 745 clients falling into this category.

**Chart 1**

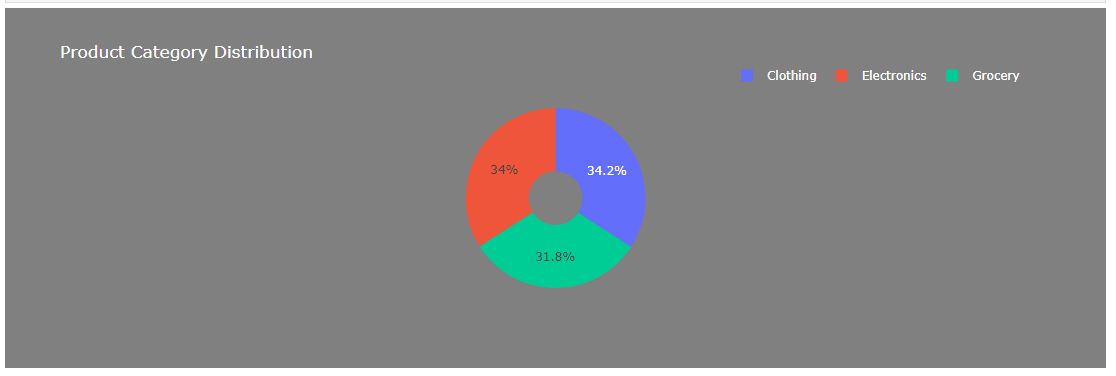


The results show that neither conventional gender, male or female are overrepresented in the client base. The fact that the others category is the most dominant type of client indicates most of Numpy Supermarket’s clients are either relatively young or progressive to identify as other. Furthermore, all three categories fall within the same range of values, all north of 740 with no category forming an out-right majority of the client base.

# Product Category Distribution

Our data further showed that the most popular product category was clothing, which represented 34,2% of the total products sold. When this evaluated in relation to the gender classifications, it is plausible that the reason we see a higher demand for electronics is influenced by the type of client who frequents Numpy Supermarket the most, the young client. Like the results on gender classification, all major product categories fall within the same range of distribution, all over 30% with the second most popular product category clothing representing 34% of all products purchased. Meanwhile the groceries categories represent 31.8% of the products sold by Numpy Supermarket. **Chart 2** below summarizes these findings.

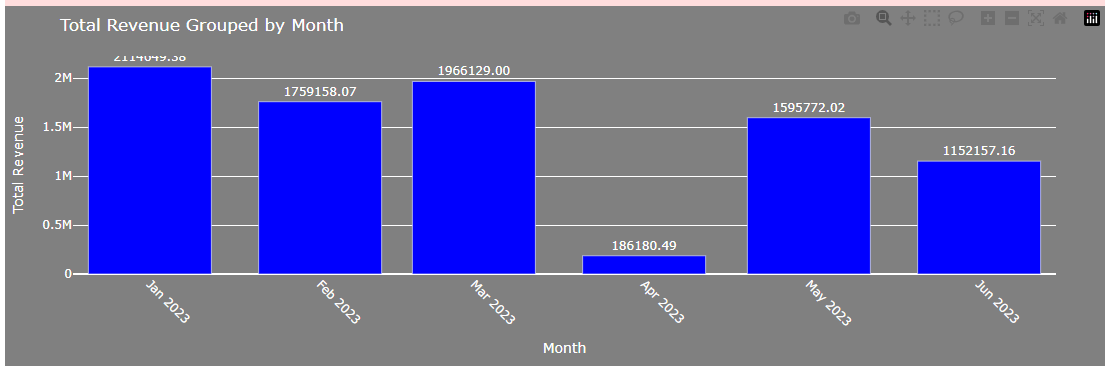
**Chart 2**



# Total Revenue per month

The data shows that January 2023 was the top month in terms of revenue with N$2.11 million worth of sales recorded for that month (see **Chart 3** below). March 2023 was the second-best month for sales as N$1.97 million worth of sales were recorded during that month. February 2023 closes of the top three with N$1.76 million worth of sales. Numpy Supermarket will have to investigate the drastic drop in sales during the April as only N$186,18 thousand was recorded in revenue.

**Chart 3**



# Machine learning

# conclusion