Amazon Product Performance Analysis

Insights through Web Scraping, Data Transformation, and Visualization

---

Table Of Contents

1. Project Overview

This project demonstrates the use of Python for web scraping, Power Query for data transformation, and Power BI for data Modeling and visualization. The goal is to extract product data from Amazon, clean and transform it, and then perform analysis to gain insights into product performance.

2. Data Collection: Web Scraping with Python

For the data collection phase, I used Python's BeautifulSoup library to extract product details from an Amazon webpage. The webpage data was read from an HTML file, and specific product details such as name, price, reviews, quantity, ratings, and original price were extracted and saved into a CSV file.

Python Code for Web Scraping:

```
from bs4 import BeautifulSoup

import pandas as pd


# Read and parse the HTML file

with open(r"D:\DATA ANALYTICS\python\Amazon_data.html",'r',encoding='utf8') as
file:

    soup = BeautifulSoup (file, 'html.parser')


# Lists to store the extracted data

product_names = []

product_prices = []

product_reviews = []

quantity_for_pastmonth = []

ratings = []

original_price = []


# Find all product containers

divs = soup.find_all('div', class_="sg-col-inner")

print(f"Found {len(divs)} product divs")  # Debugging line


# Extract details for each product

for div in divs:

    # Product Name

    name_element = div.find('span' , class_="a-size-base-plus a-color-base a-text-
normal")

    product_name = name_element.get_text(strip=True) if name_element else " "

    product_names.append(product_name)
```

```
    # Product Price
    price_element = div.find('span', class_="a-price-whole")
    product_price = price_element.get_text(strip=True) if price_element else " "
    product_prices.append(product_price)

    # Product Reviews
    review_element = div.find('span', class_="a-size-base")
    product_reviews_text = review_element.get_text(strip=True) if review_element
else " "
    product_reviews.append(product_reviews_text)

    # Quantity for past month
    quantity_element = div.find('span', class_="a-size-base a-color-secondary")
    product_quantity_text = quantity_element.get_text(strip=True) if
quantity_element else " "
    quantity_for_pastmonth.append(product_quantity_text)

    # Ratings
    rating_element = div.find('span', class_="a-icon-alt")
    product_rating_text = rating_element.get_text(strip=True) if rating_element
else " "
    ratings.append(product_rating_text)

    # Original Price
    originalprice = div.find_all('span', class_="a-offscreen")
    originalprice_element = originalprice[1] if len(originalprice) > 1 else None
    product_original_text = originalprice_element.get_text(strip=True) if
originalprice_element else " "
    original_price.append(product_original_text)

# Create a DataFrame and save to CSV
data = {
    'Product_Name': product_names,
    'Product_Price': product_prices,
```

```
    'Product_Reviews': product_reviews,
    'Product_Quantity': quantity_for_pastmonth,
    'Product_Ratings': ratings,
    'Original_Price': original_price
}

df = pd.DataFrame(data)
output_path = r'D:\DATA ANALYTICS\Amazon_Product_Data\Amazon_data.csv'
df.to_csv(output_path, index=False)
print(f"Data saved to {output_path}")
```

Explanation of Data Collected:

- **Product Name:** Extracted using the class `a-size-base-plus a-color-base a-text-normal`.
- **Product Price:** Extracted using the class `a-price-whole`.
- **Product Reviews:** Extracted from the `a-size-base class`.
- **Quantity for Past Month:** Extracted from the `a-size-base a-color-secondary class`.
- **Ratings:** Extracted from the `a-icon-alt` class.
- **Original Price:** Extracted from the `a-offscreen` class.

3. Data Transformation: Power Query and Excel

After collecting the data, I used Excel and Power Query to clean and transform it for analysis.

Key Transformation Steps:

- **Replacing Blank Rows:** Filtered empty cells and replaced them with 0.
- **Flash Fill:** Used Flash Fill to standardize data columns.
- **Price Differences:** Calculated the difference between the original price and the selling price to determine the discount value.

Power Query M Code for Converting Number Formats:

The following Power Query M code was used to transform text-based purchase data (e.g., "1.2K" or "50+") into numerical values:

```
= Table.AddColumn(#"Inserted Text Range", "Custom", each
    if Text.Contains([Text Range], "K") then
        Number.From(Text.Replace(Text.Split([Text Range], "+"){0}, "K", "")) *
1000
    else if Text.Contains([Text Range], "+") then
        Number.From(Text.Replace([Text Range], "+", ""))
    else
        Number.From([Text Range])
)
```

4. Data Modeling: Power BI DAX Functions

After transforming the data, I used Power BI to create a data model and calculate metrics using DAX.

Key DAX Formulas Used:

- **Average Discount:**

```
AVERAGE DISCOUNT = CALCULATE(AVERAGE(Data[Discount Price]),
FILTER(ALL(Data), Data[Discount Price] > 0))
```

- **Average Discount Percentage:**

```
AVERAGE DISCOUNT PERCENTAGE = CALCULATE(AVERAGE(Data[Discount Percentage]),
FILTER(ALL(Data), Data[Discount Percentage] > 0))
```

- **Average Products Purchased (Last Month):**

```
AVG. PRODUCTS PURCHASED (LAST MONTH) = CALCULATE(AVERAGE(Data[Products
Purchased]), ALL(Data))
```

- **Selected Value (for displaying product name):**

```
SELECTEDVALUE(Data[Product Name])
```

- **Parameter for Category and Sub-Category**

```
Parameter = {
  ("Category", NAMEOF('Data'[Category]), 0),
 ("Sub-Category", NAMEOF('Data'[Sub-Category]), 1)
}
```

These DAX formulas enabled me to perform in-depth analysis, providing the flexibility to categorize products, calculate discounts, and track product purchases dynamically.

5. Data Visualization: Power BI

This section highlights the visual insights generated using Power BI.

Visualization 1: Top 10 Most Purchased Products

The Stacked Bar Chart highlights the most purchased products, giving insights into product popularity.

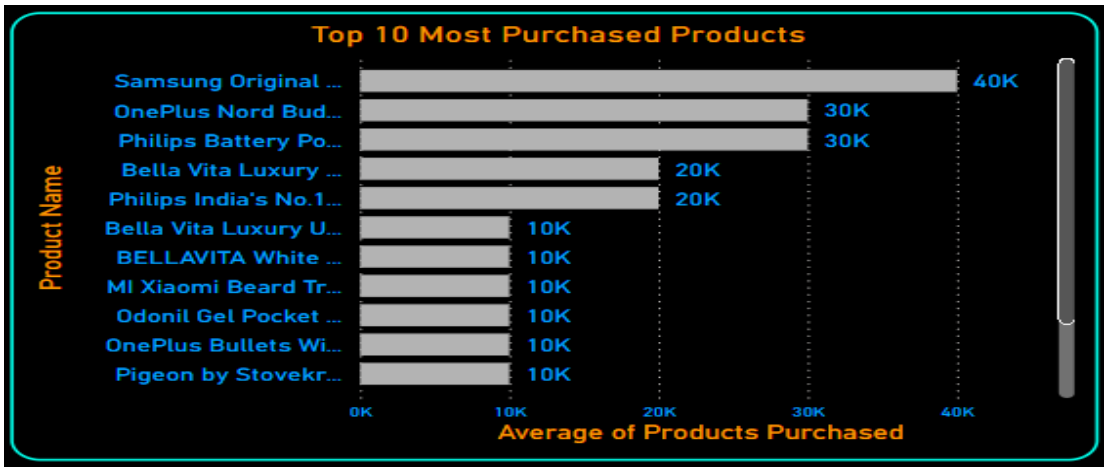

Figure 1

Visualization 2: Average of Products Purchased in Past Month

The Stacked Column Chart illustrates the average number of products purchased in the past month, broken down by category.

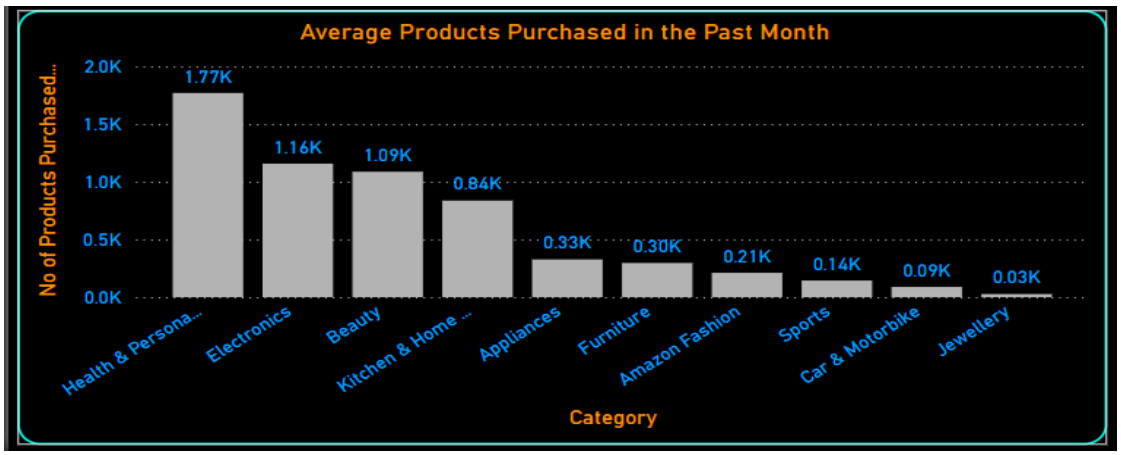

Figure 2

Visualization 3: Ratings Above 4 by Category

The Pie Chart helps identify the distribution of products with ratings above 4 in different categories.

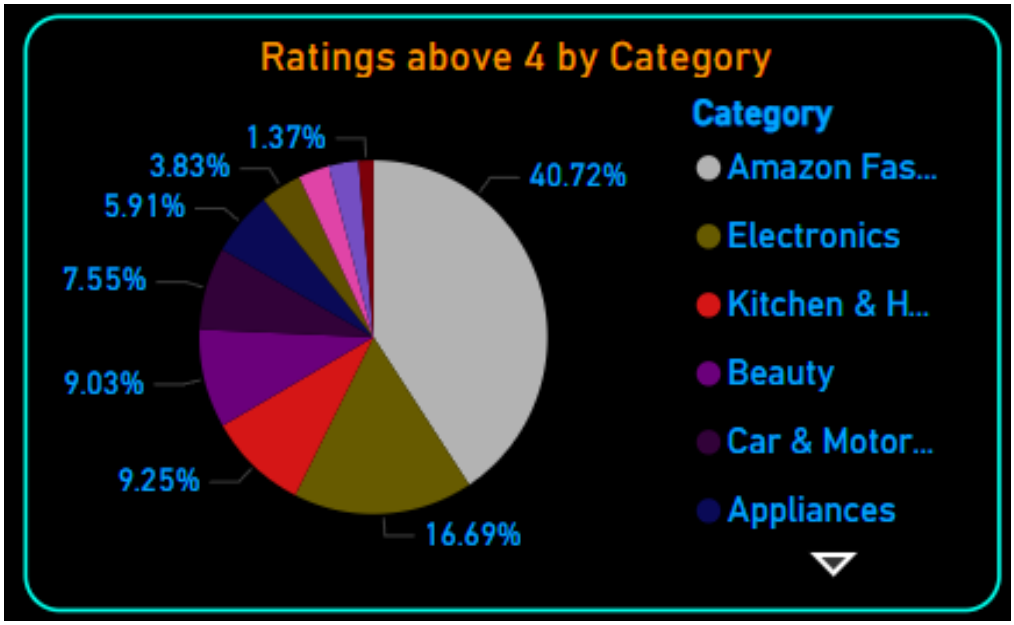

Figure 3

6. Summary of Key Insights

The analysis provided the following insights:

- **Discount Trends:** Products with higher discounts often had lower original prices.
- **Product Popularity:** Highly purchased products had better ratings and reviews.
- **Pricing Strategies:** Significant discounting trends were observed, with selling prices often much lower than original prices.

7. Conclusion

This project demonstrates the use of an end-to-end data analysis pipeline. Using Python for web scraping, Power Query for data cleaning, and Power BI for Modeling and visualization, we derived actionable insights into product performance.