Table of Contents

Chapter One | Understand the Data 📊

[Table of Content 📜⬆️]

In [1]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns

import missingno as msno
from tabulate import tabulate
from wordcloud import WordCloud

import plotly.express as px
import plotly.graph_objects as go
import plotly.figure_factory as ff
from plotly.subplots import make_subplots

import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
source = r'C:\Users\Rudra\Downloads\Python_Pandas_profiling\Phone_Search\phc
df = pd.read_csv(source, encoding= 'unicode_escape')
df.sample(3)
```

Out[2]:

	asin	product_title	product_price	product_original_price	currenc
196	B0CKXNCDXY	CAT S62 Rugged Cell Phone Unlocked (128GB, 4GB...	$169.00	NaN	US
2	B09SM24S8C	Samsung Galaxy A03s Cell Phone, AT&T GSM U...	$69.00	$99.99	US
206	B0D7SBL8PJ	X40 Unlocked 5G Cell Phones 2024 Android 13 Mo...	$99.99	NaN	US

3 rows × 22 columns

In [3]:
```
print("The size of the DataFrame:",df.size, '\n')
print('The Shape of the DataFrame', df.shape, '\n')
print("Available columns in the DataFrame:", df.columns)
```

```
The size of the DataFrame: 7480

The Shape of the DataFrame (340, 22)

Available columns in the DataFrame: Index(['asin', 'product_title', 'product
_price', 'product_original_price',
       'currency', 'product_star_rating', 'product_num_ratings', 'product_ur
l',
       'product_photo', 'product_num_offers', 'product_minimum_offer_price',
       'is_best_seller', 'is_amazon_choice', 'is_prime',
       'product_availability', 'climate_pledge_friendly', 'sales_volume',
       'delivery', 'has_variations', 'unit_price', 'unit_count',
       'coupon_text'],
      dtype='object')
```

In [4]:
```
# info summary
info = {
    "Index": df.index,
    "Columns": df.columns.tolist(),
    "Non-Null Count": df.notnull().sum().tolist(),
    "Dtype": df.dtypes.tolist()
}

# Convert to a format suitable for tabulation
info_table = zip(info["Columns"], info["Non-Null Count"], info["Dtype"])

# Print the summary information in a table format
print(tabulate(info_table, headers=["Column", "Non-Null Count", "Dtype"], ta
```

Column	Non-Null Count	Dtype
asin	340	object
product_title	340	object
product_price	336	object
product_original_price	148	object
currency	336	object
product_star_rating	337	float64
product_num_ratings	340	int64
product_url	340	object
product_photo	340	object
product_num_offers	340	int64
product_minimum_offer_price	336	object
is_best_seller	340	bool
is_amazon_choice	340	bool
is_prime	340	bool
product_availability	75	object
climate_pledge_friendly	340	bool
sales_volume	320	object
delivery	337	object
has_variations	340	bool
unit_price	4	object
unit_count	4	float64
coupon_text	22	object

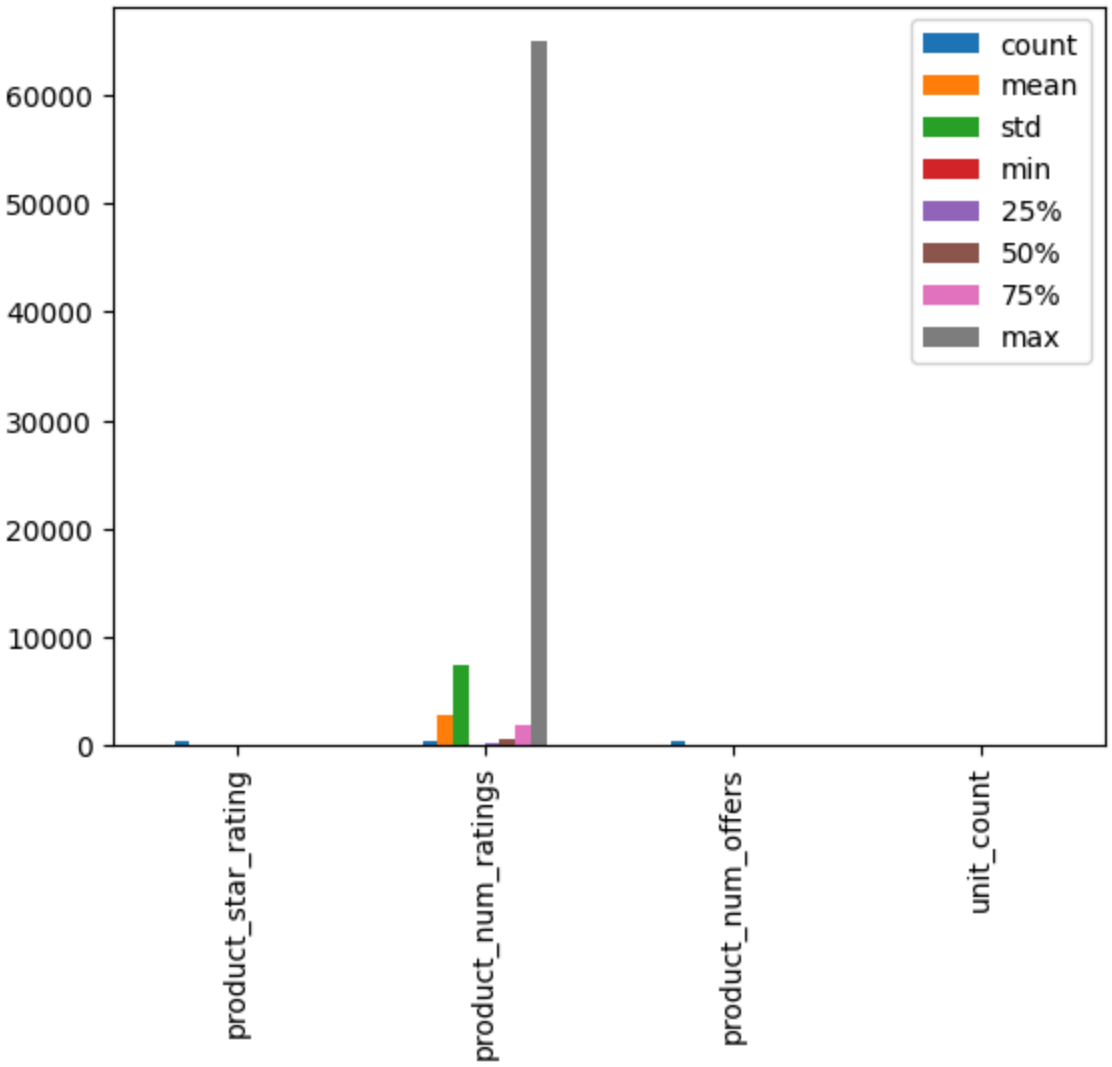
In [5]:
```
df.describe(include='all').T
```

Out[5]:

	count	unique	
asin	340	315	B0BQ11
product_title	340	315	Moto G Play 2023 3-Day Battery Unlo Mad
product_price	336	241	$9
product_original_price	148	108	$11
currency	336	1	
product_star_rating	337.0	NaN	
product_num_ratings	340.0	NaN	
product_url	340	315	https://www.amazon.com/dp/B0BQ11
product_photo	340	304	https://m.m amazon.com/images/I/71zGrrAe
product_num_offers	340.0	NaN	
product_minimum_offer_price	336	273	$9
is_best_seller	340	2	I
is_amazon_choice	340	2	I
is_prime	340	2	
product_availability	75	21	Only 1 left in stock - order s
climate_pledge_friendly	340	2	I
sales_volume	320	18	100+ bought in past m
delivery	337	92	FREE delivery Tue, Se
has_variations	340	2	I
unit_price	4	2	$2
unit_count	4.0	NaN	
coupon_text	22	13	Save 10% with cou

In [6]:
```
df.describe().T.plot(kind='bar')
```

Out[6]:
```
<Axes: >
```



Chapter Two | Data Cleaning 🧹

[Table of Content 📜⬆️]

In [7]:
```
df.duplicated().sum()
```

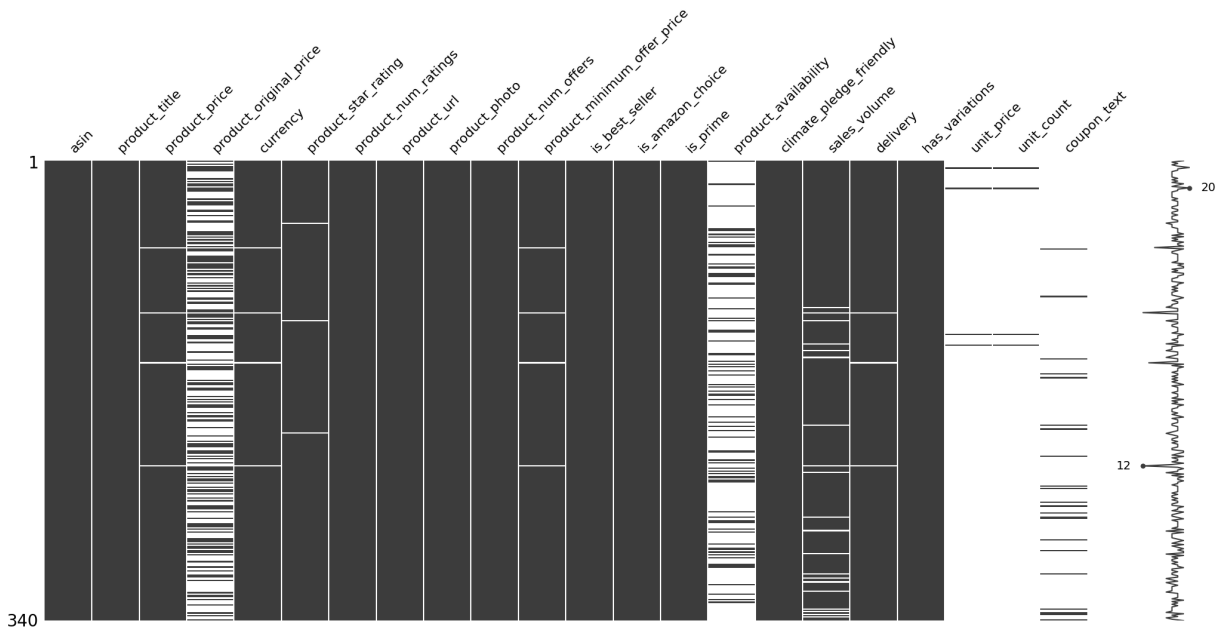
Out[7]: 21

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
asin                            0
product_title                   0
product_price                   4
product_original_price        192
currency                        4
product_star_rating             3
product_num_ratings             0
product_url                     0
product_photo                   0
product_num_offers              0
product_minimum_offer_price     4
is_best_seller                  0
is_amazon_choice                0
is_prime                        0
product_availability          265
climate_pledge_friendly         0
sales_volume                   20
delivery                        3
has_variations                  0
unit_price                    336
unit_count                    336
coupon_text                   318
dtype: int64
```

In [9]:
```
msno.matrix(df)
```

Out[9]: <Axes: >



⭐ **Best Practice before clean the data have a copy of Original Data**

1. Remove the duplicated data
2. Delete these columns (unit_price, unit_count,

coupon_text,product_original_price,product_avail
)
3. currency, product_url, product_photo, Deleted no sense for analysis
4. Change the Price & product_minimum_offer_price into int

[Table of Content 📜⬆️]

In [11]:
```
dfc = df.copy()
```

In [12]:
```
dfc.sample()
```

Out[12]:

	asin	product_title	product_price	product_original_price	currency
295	B00ZYHA1KI	Hamilton CapTel 2400i Captioned Telephone Larg...	$75.00	NaN	USD

1 rows × 22 columns

In [13]:
```
dfc['currency'].unique() #All values are USD or nan so remove it.
```

Out[13]:
```
array(['USD', nan], dtype=object)
```

In [14]:
```
dfc.columns
```

Out[14]:
```
Index(['asin', 'product_title', 'product_price', 'product_original_price',
       'currency', 'product_star_rating', 'product_num_ratings', 'product_u
rl',
       'product_photo', 'product_num_offers', 'product_minimum_offer_pric
e',
       'is_best_seller', 'is_amazon_choice', 'is_prime',
       'product_availability', 'climate_pledge_friendly', 'sales_volume',
       'delivery', 'has_variations', 'unit_price', 'unit_count',
       'coupon_text'],
      dtype='object')
```

In [15]:
```
# drop the columns
dfc.drop(columns=['currency', 'product_original_price','unit_price', 'unit_c
```

In [16]:
```
# Change the Datatypes
dfc['product_minimum_offer_price'] = dfc['product_minimum_offer_price'].str.
dfc['product_price'] = dfc['product_price'].str.replace('$', '').str.replace
```

In [17]:
```
dfc[['sales_volume',    'delivery']].sample(3)
```

Out[17]:

	sales_volume	delivery
292	50+ bought in past month	FREE delivery Tue, Sep 24
161	100+ bought in past month	FREE delivery Tue, Sep 24
105	1K+ bought in past month	FREE delivery Tue, Sep 24 on $35 of items ship...

Chapter Three | Analysis 🔍

[Table of Content 📜⬆️]

⚡ 1. Correlation Matrix

In [18]:
```
numeric_df = dfc.select_dtypes(include=[float, int])
correlation_matrix = numeric_df.corr().T

fig = px.imshow(correlation_matrix, text_auto=True, aspect="auto",
                title='Correlation Matrix',
                color_continuous_scale='viridis')
fig.show()
```

⚡ 2. Star Rating with Price Comparisons

In [19]:
```
# Create subplots with 2 rows and 1 column
fig = make_subplots(rows=2, cols=1, subplot_titles=['Star Rating vs. Product

# First histogram for 'product_star_rating' vs. 'product_price'
hist1 = px.histogram(dfc, x='product_star_rating', y='product_price')
fig.add_trace(hist1.data[0], row=1, col=1)

# Second histogram for 'product_star_rating' vs. 'product_minimum_offer_pric
hist2 = px.histogram(dfc, x='product_star_rating', y='product_minimum_offer_
fig.add_trace(hist2.data[0], row=2, col=1)
```

```
# Update the layout and show the figure
fig.update_layout(title_text='Star Rating with Price Comparisons', height=80
fig.show()
```

⚡ 3.Rating and Price Relationship

In [20]:
```
fig = px.scatter_ternary(
    dfc,
    a=dfc['product_price'],
    b=dfc['product_num_ratings'],
    c=dfc['product_star_rating'],
    color=dfc['has_variations'],
    title='Rating and Price Relationship',

)

fig.show()
```

⚡4. Offers and Price Relationship

In [21]:
```
import plotly.express as px

# Create a ternary scatter plot
fig = px.scatter_ternary(
    dfc,
    a='product_price',
    b=dfc['product_num_offers'],
    c=dfc['product_minimum_offer_price'],
    size='product_num_ratings',
    size_max=60,
    color='has_variations',
    title='Offers and Price Relationship'
)

# Display the figure
fig.show()
```

⚡ 5. Different Field Analysis

In [22]:
```
# Create subplots with 3 rows and 2 columns
fig = make_subplots(
    rows=3,
    cols=2,
    subplot_titles=('Amazon Choice', 'Best Seller', 'Prime Member', 'Climate
    specs=[[{'type': 'pie'}, {'type': 'pie'}],
           [{'type': 'pie'}, {'type': 'pie'}],
           [{'type': 'pie'}, None]]
)

# Add a pie chart for 'is_amazon_choice' in the first subplot
fig.add_trace(
    go.Pie(labels=dfc['is_amazon_choice'].value_counts().index,
           values=dfc['is_amazon_choice'].value_counts().values,
           name='Amazon Choice', rotation=50),
    row=1,
    col=1
)

# Add a pie chart for 'is_best_seller' in the second subplot
fig.add_trace(
    go.Pie(labels=dfc['is_best_seller'].value_counts().index,
           values=dfc['is_best_seller'].value_counts().values,
           name='Best Seller', rotation=50),
    row=1,
    col=2
)

# Add a pie chart for 'is_prime' in the third subplot
fig.add_trace(
    go.Pie(labels=dfc['is_prime'].value_counts().index,
           values=dfc['is_prime'].value_counts().values,
           name='Prime Member'),
    row=2,
    col=1
)

# Add a pie chart for 'climate_pledge_friendly' in the fourth subplot
fig.add_trace(
    go.Pie(labels=dfc['climate_pledge_friendly'].value_counts().index,
           values=dfc['climate_pledge_friendly'].value_counts().values,
           name='Climate Friendly'),
    row=2,
    col=2
)

# Add a pie chart for 'has_variations' in the fifth subplot
fig.add_trace(
    go.Pie(labels=dfc['has_variations'].value_counts().index,
           values=dfc['has_variations'].value_counts().values,
           name='Has Variations'),
```

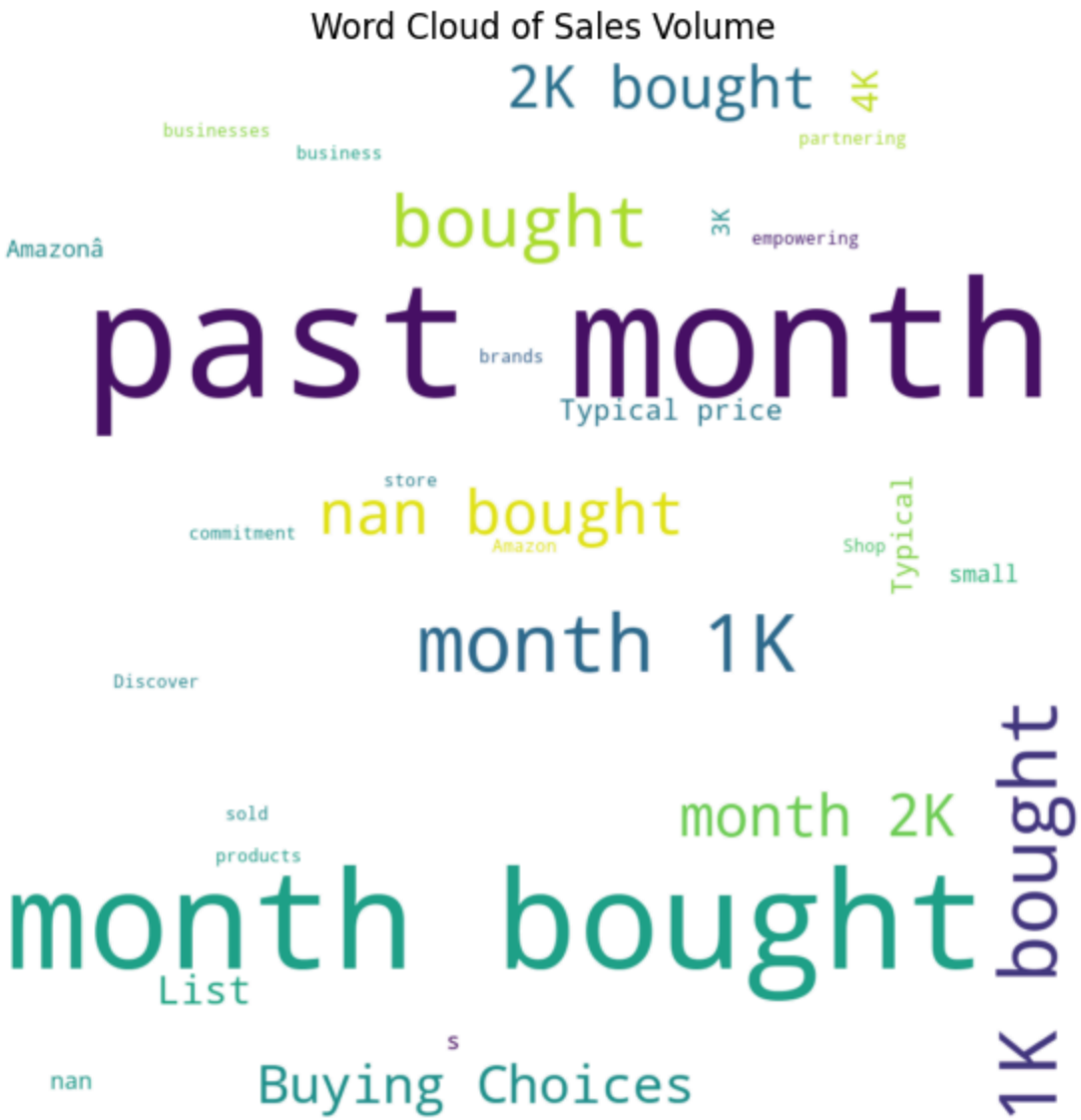
```
    row=3,
    col=1
)

# Update layout for better visualization
fig.update_layout(
    title_text='Different Field Analysis',
    height=1000,
    width=1200
)

# Display the figure
fig.show()
```

⚡ 6.Word Cloud of Delivery

In [23]:
```
# Create a string of text
text = ' '.join(dfc['delivery'].astype(str) )

# Generate the word cloud
wordcloud = WordCloud(width=1000, height=1000, background_color='white').ger

# Display the word cloud
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')  # Turn off axis
plt.title("Word Cloud of Delivary")
plt.show()
```

⚡ 7.Word Cloud of Sales volume

In [24]:
```
# Create a string of text
text = ' '.join(dfc['sales_volume'].astype(str) )

# Generate the word cloud
wordcloud = WordCloud(width=1000, height=1000, background_color='white').ger

# Display the word cloud
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')  # Turn off axis
plt.title("Word Cloud of Sales Volume")
plt.show()
```



Last but not least, the final chapter

| Draw an Conclusion 📝

[Table of Content 📜⬆️]

Conclusion

1. **Product Rating:** The majority of products on the platform have higher ratings, indicating a generally positive customer experience.

2. **Price Trends:** During offer periods, there is a noticeable drop in product prices, making it an ideal time for customers to make purchases.
3. **Amazon Choice and Best Seller Products:** Both Amazon Choice and Best Seller products represent a small portion of the total, indicating limited availability or selection under these categories.
4. **Prime Membership:** Approximately **27%** of the customers are Prime members, suggesting a moderate uptake of the membership benefits.
5. **Best Sellers:** Less than **1%** of the products are marked as Best Sellers on Amazon, showing the exclusivity of this label.
6. **Product Variations:** A significant **70.6%** of products come with variations, such as different sizes, colors, or models, offering customers more options.
7. **Climate Consciousness:** Around **77%** of customers prefer climate-friendly products, highlighting a growing awareness and preference for sustainable options.
8. **Customer Attraction:** The term "FREE DELIVERY" appears frequently, indicating that customers are highly attracted to products that offer free shipping.

These conclusions provide insights into customer preferences, product trends, and market dynamics on the platform.