
Report : Customer Segmentation – Task 2

INTELLIHACK 5.0

Team: Pandas and Pythons

Introduction

In today's competitive digital marketplace, understanding customer behavior is crucial for businesses to optimize their marketing strategies, enhance user experience, and increase revenue. Customer segmentation plays a vital role in identifying distinct consumer groups based on purchasing patterns, browsing behaviors, and discount utilization. This report focuses on an in-depth exploratory data analysis (EDA) and clustering-based segmentation of customer data to uncover hidden patterns. By leveraging advanced statistical techniques and machine learning models, we aim to classify customers into meaningful segments that provide actionable insights for business decision-making.

Scope

This analysis is centered around a dataset containing key behavioral metrics such as total purchases, average cart value, time spent on the platform, product clicks, and discount usage. The primary objective is to segment customers into three distinct groups: **Bargain Hunters, High Spenders, and Window Shoppers**. The study involves a comprehensive EDA to identify patterns, feature engineering to enhance model performance, and clustering algorithms to define customer segments. Additionally, evaluation metrics and visualization techniques are used to validate the clustering results. The findings from this study can assist businesses in tailoring personalized marketing strategies, optimizing promotions, and improving customer retention.

1. Exploratory Data Analysis (EDA)

1.1 Reading the Dataset

The dataset is loaded using pandas and contains customer behavior metrics. The first and last few rows are displayed to verify data integrity.

```
[2] # Read the dataset
    df = pd.read_csv('drive/MyDrive/Intellihack_Pandas_and_Pythons_Task_2/customer_behavior_analytcis.csv')

    df.head()
```

	total_purchases	avg_cart_value	total_time_spent	product_click	discount_counts	customer_id
0	7.0	129.34	52.17	18.0	0.0	CM00000
1	22.0	24.18	9.19	15.0	7.0	CM00001
2	2.0	32.18	90.69	50.0	2.0	CM00002
3	25.0	26.85	11.22	16.0	10.0	CM00003
4	7.0	125.45	34.19	30.0	3.0	CM00004

```
    df.tail()
```

	total_purchases	avg_cart_value	total_time_spent	product_click	discount_counts	customer_id
994	5.0	64.64	72.70	50.0	1.0	CM00994
995	5.0	68.36	75.41	43.0	1.0	CM00995
996	18.0	19.53	28.77	18.0	8.0	CM00996
997	4.0	28.97	72.27	57.0	3.0	CM00997
998	29.0	39.29	9.99	16.0	11.0	CM00998

1.2 Basic Information & Sanity Check

To understand the dataset's structure, we check column types, missing values, and overall shape.

A summary of key statistical measures helps us understand data distribution. This provides insights into **mean, standard deviation, min/max values**, and percentiles for each numerical feature. Distributions of key features like total_purchases, avg_cart_value, and total_time_spent can indicate variability across customers.

```
# Descriptive Statistics
df.describe()
```

	total_purchases	avg_cart_value	total_time_spent	product_click	discount_counts
count	979.000000	979.000000	999.000000	979.000000	999.000000
mean	11.570991	75.457978	49.348759	28.237998	4.313313
std	7.016327	55.067835	32.730973	16.296384	4.532772
min	0.000000	10.260000	5.120000	4.000000	0.000000
25%	6.000000	33.130000	22.375000	16.000000	1.000000
50%	10.000000	49.380000	40.360000	21.000000	2.000000
75%	17.000000	121.255000	77.170000	45.000000	8.000000
max	32.000000	199.770000	119.820000	73.000000	21.000000

```
# Display basic information - Sanity Check
df.info()

df.shape
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 999 entries, 0 to 998
Data columns (total 6 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   total_purchases   979 non-null    float64
 1   avg_cart_value    979 non-null    float64
 2   total_time_spent  999 non-null    float64
 3   product_click     979 non-null    float64
 4   discount_counts   999 non-null    float64
 5   customer_id       999 non-null    object
dtypes: float64(5), object(1)
memory usage: 47.0+ KB
(999, 6)
```

1.3 Feature Distribution Analysis

To analyze categorical features, we use value_counts() to check unique values and their frequency.

```
# Check for object columns to identify the data distribution
for i in df.select_dtypes(include='object').columns:
  print(df[i].value_counts())
  print("***" * 10)
```

For numerical features, value_counts() shows distinct values and their occurrences.

```
# Check for number columns to identify the data distribution
for i in df.select_dtypes(include='number').columns:
  print(df[i].value_counts())
  print("***" * 10)
```

1.4 Data Quality Checks

To ensure data integrity, we check for duplicate entries: There were **no any duplicates**.

```
# Check for Duplicates
print("Duplicate Rows:", df.duplicated().sum())
```

```
Duplicate Rows: 0
```

Missing values can affect clustering performance, so we identify and visualize them: There were **20 null valued rows.**

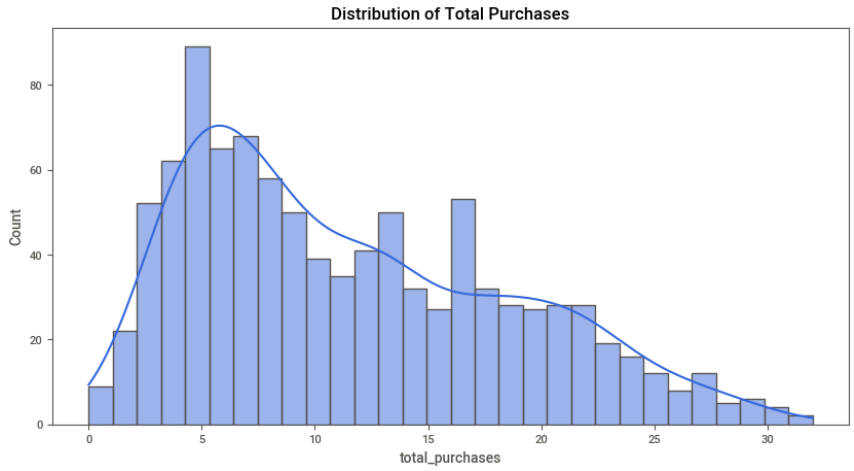
```
# Check for missing values
print("Missing Values:\n", df.isnull().sum())
```

```
Missing Values:
 total_purchases     20
avg_cart_value       20
total_time_spent      0
product_click        20
discount_counts       0
customer_id           0
dtype: int64
```

A heatmap helps visualize missing values:

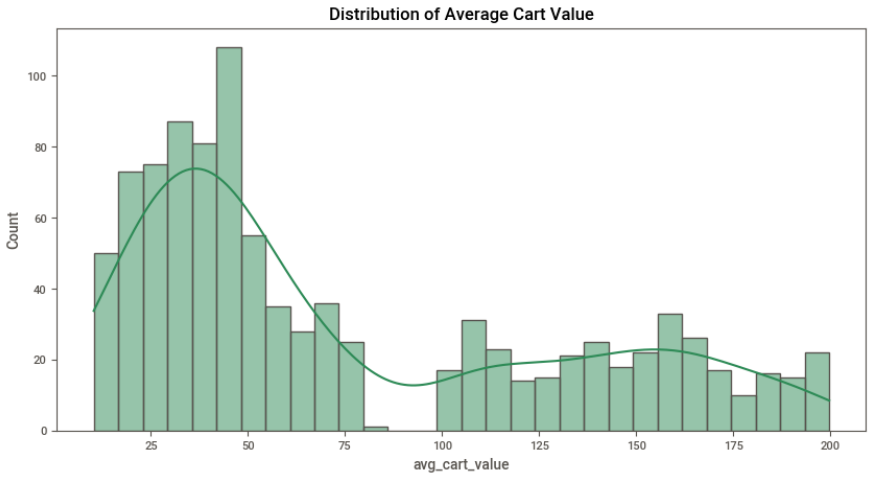
1.5 Univariate Analysis

1. total_purchases (Total Number of Purchases)



Right-skewed, most customers make very few purchases, meaning the platform may rely on a small number of high-value customers

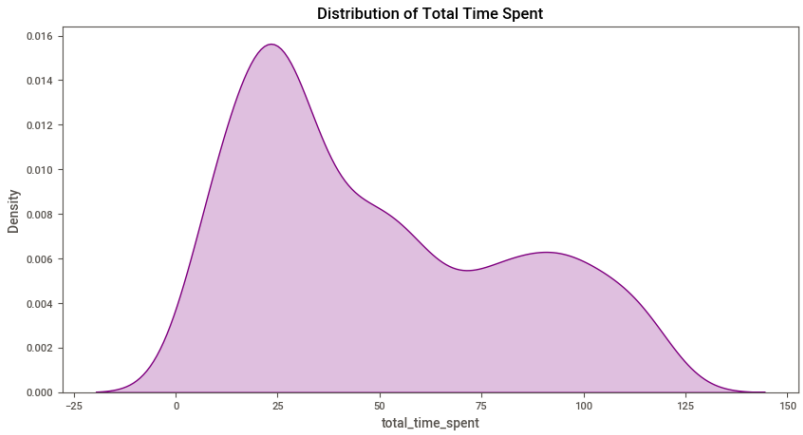
2. avg_cart_value (Average Cart Value per Purchase)



Bimodal, there may be two groups: budget buyers vs. premium buyers

Somewhat right-skewed, the platform might have a lower high-value spenders driving revenue.

3. total_time_spent (Total Minutes on Platform)

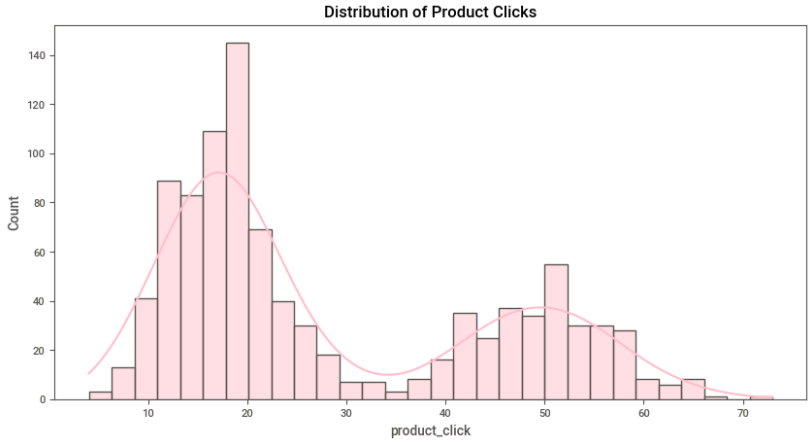


If some users spend a lot of time but don't purchase, they may need better product recommendations.

If high time = high purchases, time spent is a key factor in revenue.

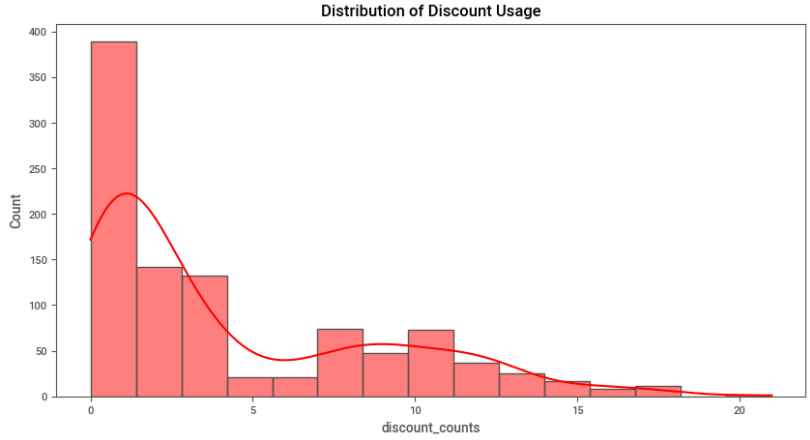
4. product_click (Number of Products Viewed)

Bimodal, there may be two groups according to the product clicking: Usually viewing less number of products, and view many products and compare each other with.



This can be due to some people are premium buyers, who don't look much into many products, and budget buyers, who much look into many products.

5. discount_count (Number of Discounts Used)



There may be customers **only buy with discounts**, revenue might depend too much on promotions

If some **high-spenders don't use discounts**, they may be premium users.

1.6 Bi-variate Analysis

1. Total Purchases vs. Average Cart Value

This scatter plot reveals three distinct customer clusters:

1. High-Frequency, Low-Value: Customers with many purchases but low average cart values (likely "Bargain Hunters").
2. Moderate-Frequency, High-Value: Customers with a moderate number of purchases and high average cart values (likely "High Spenders").
3. Low-Frequency, Varied-Value: Customers with few purchases and a wide range of cart values (likely "Window Shoppers").

2. Time Spent vs. Purchase Frequency

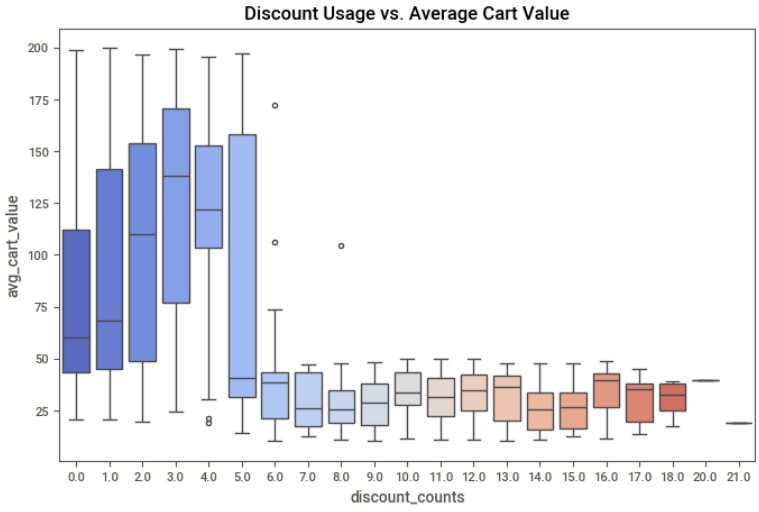
A clear negative correlation exists: longer time spent browsing correlates with fewer purchases. This indicates "Window Shoppers" (high time, low purchases) versus potential "Bargain Hunters" (low time, high purchases). However, other factors also influence buying behavior beyond just time spent.

3. Product Clicks vs. Total Purchases

- A trend emerges: higher product clicks generally correlate with lower total purchases, indicating "Window Shoppers" who browse extensively but buy less.
- Conversely, a cluster shows higher purchases with fewer clicks, potentially representing "Bargain Hunters" or "High Spenders" who quickly find desired items.
- The spread highlights varying customer behaviors regarding product exploration and purchase decisions.

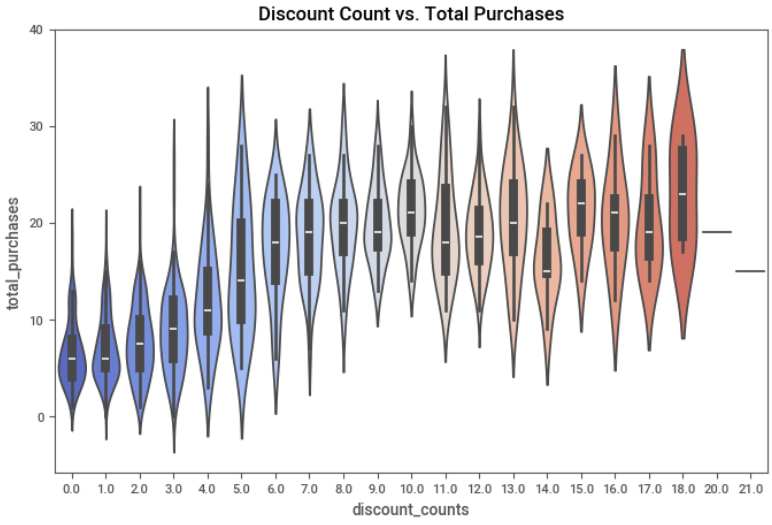
4. Discount Usage vs. Average Cart Value

- High Discount Usage, Low Cart Value: Customers with high discount usage (5+ times) consistently exhibit a low average cart value, clearly indicating "Bargain Hunters."
- Low Discount Usage, Varied Cart Value: Customers with low discount usage (0-4 times) show a wider range of average cart values, suggesting both "High Spenders" (high cart value) and "Window Shoppers" (varied cart value).
- Clear Segment Differentiation: The boxplot effectively separates the "Bargain Hunters" from other segments based on discount usage and spending patterns.



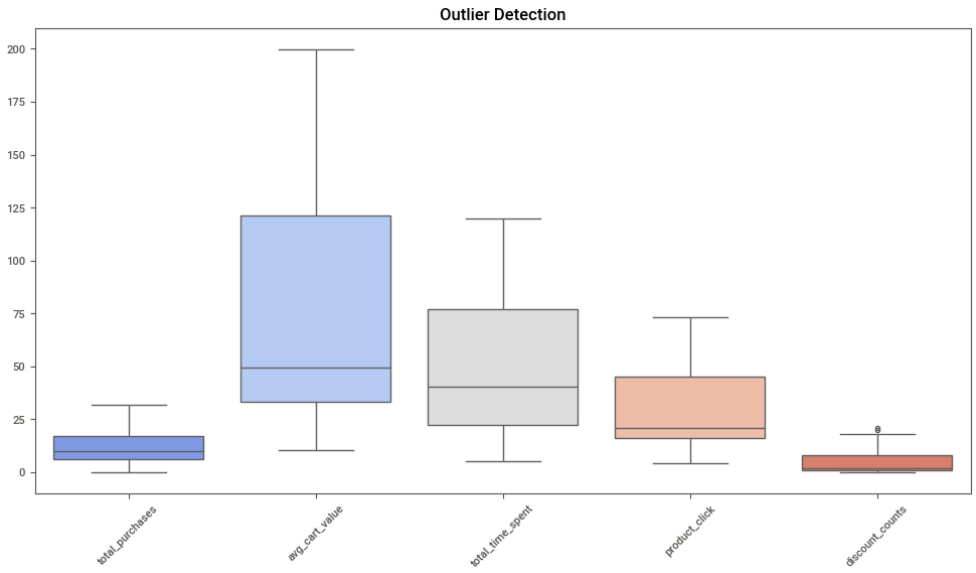
5. **Discount Count vs. Total Purchases**

- Positive Correlation: A general positive correlation exists: higher discount usage tends to correlate with higher total purchases, indicating "Bargain Hunters" who capitalize on deals.
- Distribution Shift: The violin plots show an increasing trend in total purchases as discount count increases, with a wider spread at higher discount counts, suggesting varied purchasing habits within the "Bargain Hunter" segment.
- Segment Identification: The visualization reinforces the link between high discount usage and frequent purchases, clearly identifying the "Bargain Hunters" customer segment.

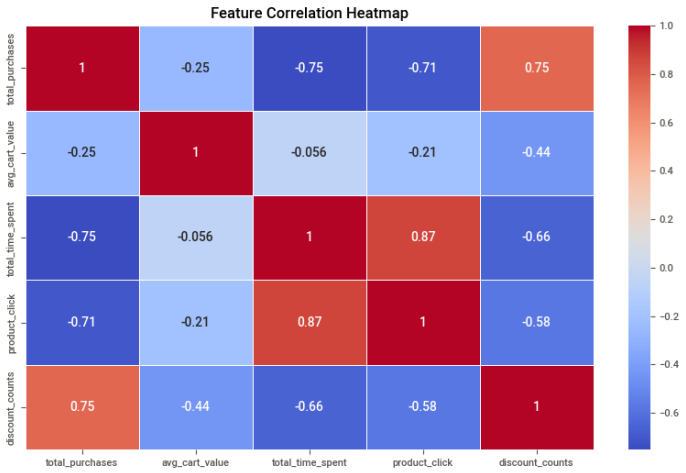


1.7 Outliers

Examination of the box plots across all features reveals no significant outliers in the dataset. This suggests that the data is relatively clean and free of extreme values that could unduly influence the clustering analysis.



1.8 Correlation Matrix



❖ **Strong Negative Correlations:**

- "total_time_spent" and "product_click" show strong negative correlations with "total_purchases," indicating that customers who spend more time or click on more products tend to make fewer purchases.
- "avg_cart_value" and "discount_counts" show a strong negative correlation, indicating that a customer who uses more discounts tends to have a lower average cart value.

❖ **Strong Positive Correlations:**

- "total_time_spent" and "product_click" exhibit a strong positive correlation, suggesting that customers who spend more time on the platform also tend to click on more products.
- "total_purchases" and "discount_counts" show a strong positive correlation, suggesting that customers who make more purchases tend to use more discount codes.

❖ **Weak Correlations:**

- "avg_cart_value" shows weak correlations with other features, implying that it is relatively independent..

1.9 Feature Scaling

Feature scaling is essential because clustering algorithms (like K-Means) rely on distance metrics (e.g., Euclidean distance). Features with larger scales (like total_purchases) can disproportionately influence the model. To address this, we scale the features using StandardScaler.

```
# Selecting relevant features
features = ['total_purchases', 'avg_cart_value', 'total_time_spent', 'product_click', 'discount_counts']

# Scaling the data
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df_2[features])
```

2. Model selection

In the customer segmentation task, selecting the appropriate clustering algorithm is key to identifying distinct customer segments (Bargain Hunters, High Spenders, Window Shoppers). Below, we explore various clustering methods and the rationale behind each.

2.1 K-Means Clustering

K-Means is a widely used algorithm for clustering numerical data. It is ideal for this problem because:

- **Scalability**: Efficient for large datasets.
- **Interpretability**: Provides clear cluster centroids, making it easy to understand segment profiles.
- **Centroid-based**: Works well for defining customer segments in terms of average behavior.

However, K-Means assumes spherical clusters, which might not always hold true for all customer behaviors, so we explore other methods.

2.2 DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

DBSCAN identifies clusters based on data density and can handle noise. It's valuable here because:

- **No need to predefine clusters**: It adapts to the data.
- **Noise handling**: It can identify outliers as noise rather than forcing them into a cluster.
- **Flexible cluster shapes**: Useful for irregular customer behavior patterns.

It requires careful tuning of parameters like eps and min_samples, but it's great for detecting clusters of varying shapes and densities.

2.3 Agglomerative Hierarchical Clustering

Agglomerative clustering builds clusters by merging the most similar ones. It's useful for:

- **Dynamic cluster number determination**: The dendrogram helps identify the optimal number of clusters.

- **Interpretability**: It shows how clusters are formed, which helps understand customer relationships.

However, it has higher computational complexity, which can be challenging for large datasets.

2.4 Gaussian Mixture Models (GMM)

GMM assumes data is generated by a mixture of Gaussians and is suitable for clusters with elliptical shapes. It offers:

- **Soft clustering**: Probabilistic membership provides a more nuanced view of customers.
- **Flexibility**: It can handle complex cluster shapes and varying densities.

GMM is computationally intensive and requires careful tuning of components and covariance types.

2.5 Comparison and Rationale

- **K-Means** is the most straightforward and efficient for well-separated clusters, making it our primary choice.
- **DBSCAN** is useful for noise handling and irregular shapes but may struggle with less dense data.
- **Hierarchical clustering** provides a clear picture of relationships between clusters but is less scalable.
- **GMM** offers flexibility but is more complex and computationally expensive.

Given the clear segmentation in customer behavior, **K-Means** is our preferred method, with other methods considered for validation.

3. Identifying the clusters

K-Means Clustering with 3 Clusters

We applied K-Means clustering with 3 clusters to the scaled dataset. The cluster centroids represent the average behavior of customers in each segment. The resulting cluster characteristics are:

- **Cluster 0 (Bargain Hunters)**: Customers in this segment have high total_purchases, low avg_cart_value, moderate total_time_spent, moderate product_click, and high discount_counts.

- **Cluster 1 (Window Shoppers)**: This group exhibits low total_purchases, high total_time_spent, high product_click, and low discount_counts.
- **Cluster 2 (High Spenders)**: Customers here have high total_purchases, high avg_cart_value, low total_time_spent, moderate product_click, and low discount_counts.

These segments clearly represent the three customer types: Bargain Hunters, Window Shoppers, and High Spenders.

Pair Plot Description and Cluster Separation:

- **Visualization:** The pair plot displays scatter plots of all feature combinations and kernel density estimations (KDEs) along the diagonal.
- **Cluster Separation:**
    - The pair plot clearly visualizes the separation of the three clusters.
    - **Total Purchases vs. Average Cart Value:** This plot shows the most distinct separation, with "Bargain Hunters" (Cluster 2) at high purchases and low cart value, "High Spenders" (Cluster 0) at moderate purchases and high cart value, and "Window Shoppers" (Cluster 1) at low purchases and varied cart value.
    - **Total Time Spent and Product Clicks:** The "Window Shoppers" (Cluster 1) are well-separated in these features, showing high time spent and product clicks.
    - **Discount Counts:** The "Bargain Hunters" (Cluster 2) are clearly identifiable by their high discount usage.
    - The KDE plots along the diagonal also illustrate the distinct distributions of each feature within each cluster.
- **Confirmation:** The pair plot confirms the effectiveness of the K-Means clustering in identifying the three customer segments.

PCA (Principal Component Analysis)

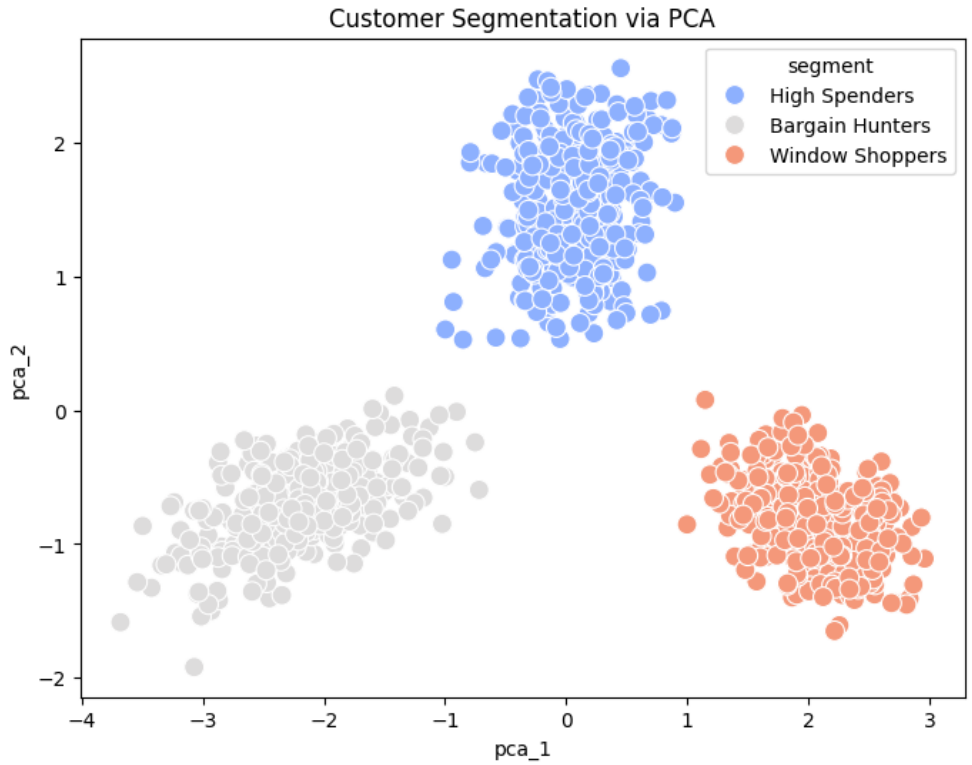
To visualize the data in a 2D space, we reduced the dataset to 2 principal components using PCA. The scatter plot below shows the customer segmentation in the 2D space of the first two principal components. The clustering results are reflected in the different colors representing the segments.

- **PCA Plot**: The scatter plot shows clear separation between the customer segments, confirming that the K-Means clustering has successfully identified distinct clusters. The PCA reduction helps visualize the complex relationships between features in a simplified 2D space.

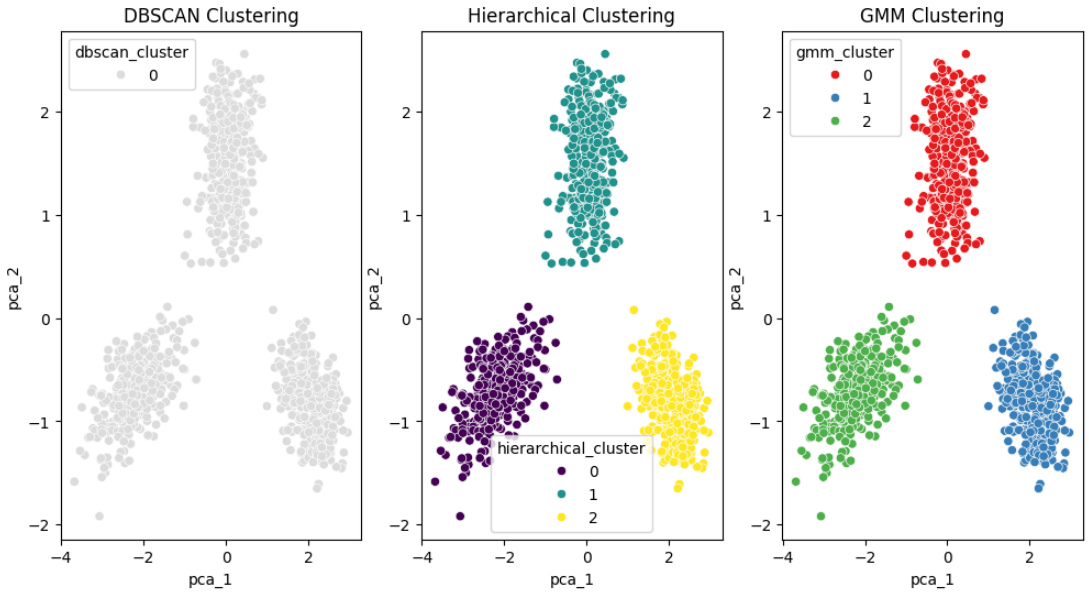
❖ **Cluster Separation:**

The plot shows clear separation between the three customer segments in the reduced PCA space.

- "High Spenders" (light blue) form a distinct cluster in the upper center of the plot.
- "Bargain Hunters" (light gray) form a distinct cluster on the left side of the plot.
- "Window Shoppers" (light red) form a distinct cluster on the right side of the plot.



Interpreting results of the other models



4. Model Evaluation

In this section, we evaluate the performance of the clustering models using various metrics to identify the best model for customer segmentation.

1. Silhouette Score

The **Silhouette Score** measures how similar each point is to its own cluster compared to other clusters. A higher score indicates better-defined clusters. The results are:

- **K-Means: 0.626**: This score indicates a moderate level of cluster cohesion and separation, suggesting that the K-Means clustering has successfully created meaningful segments.
- **DBSCAN: -1**: The negative score suggests that DBSCAN failed to form valid clusters. This might be due to its sensitivity to parameters such as eps and min_samples, leading to too many points being marked as noise.
- **Hierarchical: 0.626**: The score for hierarchical clustering is identical to K-Means, indicating that the hierarchical model produced similar results to K-Means.
- **GMM: 0.626**: Like K-Means and Hierarchical, the GMM also produces a moderate silhouette score, showing that it performed similarly to K-Means in terms of cluster cohesion and separation.

2. Davies-Bouldin Index

The **Davies-Bouldin Index** measures the average similarity ratio of each cluster with the cluster that is most similar to it. Lower values indicate better clustering. The results are:

- **K-Means: 0.55**
- **Hierarchical: 0.55**
- **GMM: 0.55**

All three models have the same Davies-Bouldin score, indicating that the intra-cluster similarity and inter-cluster separation are similar across these models. This suggests that, in terms of cluster compactness and separation, there is little difference between them.

3. Calinski-Harabasz Score

The **Calinski-Harabasz Score** evaluates the ratio of the sum of between-cluster dispersion to within-cluster dispersion. A higher score indicates better clustering. The results are:

- **K-Means: 2452.12**
- **Hierarchical: 2452.12**
- **GMM: 2452.12**

All models have the same Calinski-Harabasz score, indicating that, from the perspective of dispersion, they performed similarly in terms of defining distinct and well-separated clusters.

5. Summary and Justification

From the evaluation metrics, **K-Means** performs just as well as **Hierarchical** and **GMM** for this segmentation task. The **Silhouette Score** suggests that all three models (K-Means, Hierarchical, and GMM) created fairly well-separated clusters, while **DBSCAN** failed to form valid clusters. The **Davies-Bouldin** and **Calinski-Harabasz** scores also indicate that there is no significant difference between K-Means, Hierarchical, and GMM in terms of cluster quality.

Thus, **K-Means** is the most efficient model for this task due to its simplicity, scalability, and interpretability. Additionally, it provides clear cluster centroids, which can be easily interpreted in terms of customer behavior. Given its strong performance and computational efficiency, **K-Means** is selected as the best clustering model for customer segmentation in this case.