1. Project Description

Project Title: Customer Segmentation Using Hierarchical Clustering

Objective:
The goal of this project is to segment customers based on their purchasing behavior using hierarchical clustering, an unsupervised machine learning technique. By understanding different customer segments, businesses can tailor their marketing strategies, optimize resource allocation, and enhance customer satisfaction.

Methodology:
This project involves importing and analyzing a dataset of wholesale customer purchases. We will perform Exploratory Data Analysis (EDA) to understand the structure and relationships within the data. Hierarchical clustering will be applied to group similar customers based on their purchase patterns. Additionally, two supervised learning models will be introduced to further analyze and validate the clustering results.

2. Data Import

In [ ]:
```
import pandas as pd

# Load the dataset
df = pd.read_csv('data/Wholesale customers data.csv')

# Display the first few rows of the dataset
df.head()
```

Out[ ]:

	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
0	2	3	12669	9656	7561	214	2674	1338
1	2	3	7057	9810	9568	1762	3293	1776
2	2	3	6353	8808	7684	2405	3516	7844
3	1	3	13265	1196	4221	6404	507	1788
4	2	3	22615	5410	7198	3915	1777	5185

In [ ]:
```
df.shape
```

Out[ ]:
```
(440, 8)
```

In [ ]:
```
df.describe()
```

Out[ ]:

	Channel	Region	Fresh	Milk	Grocery	
count	440.000000	440.000000	440.000000	440.000000	440.000000	440.0
mean	1.322727	2.543182	12000.297727	5796.265909	7951.277273	3071.9
std	0.468052	0.774272	12647.328865	7380.377175	9503.162829	4854.6
min	1.000000	1.000000	3.000000	55.000000	3.000000	25.0
25%	1.000000	2.000000	3127.750000	1533.000000	2153.000000	742.2
50%	1.000000	3.000000	8504.000000	3627.000000	4755.500000	1526.0
75%	2.000000	3.000000	16933.750000	7190.250000	10655.750000	3554.2
max	2.000000	3.000000	112151.000000	73498.000000	92780.000000	60869.0

3. Identify the Machine Learning Problem

Problem Type:
This is an unsupervised learning problem focused on customer segmentation. The goal is to group customers into clusters based on their purchasing behavior without predefined labels.

Chosen Algorithm:
Hierarchical Clustering is selected for this project due to its ability to create a hierarchy of clusters, which can be visualized using a dendrogram. This technique is useful for understanding the relationships between clusters at different levels of granularity.

4. Exploratory Data Analysis (EDA)

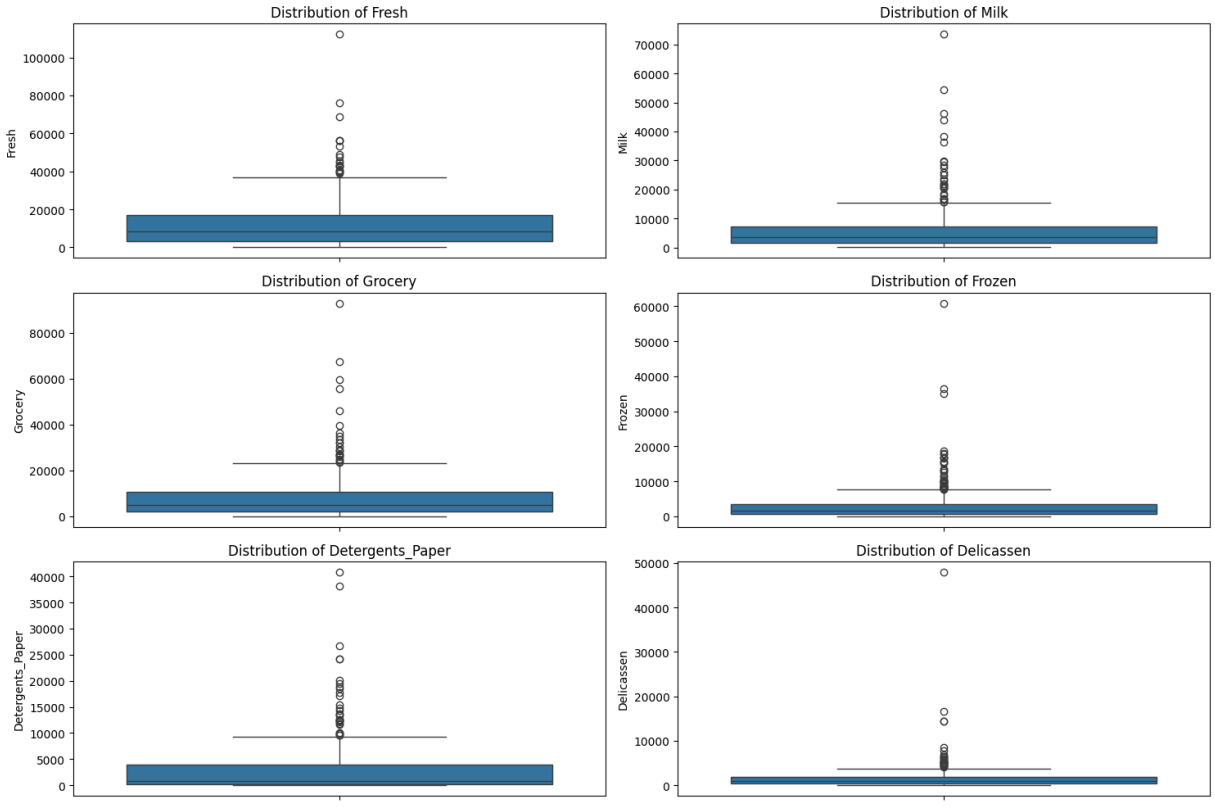
4.1 Describe the Features

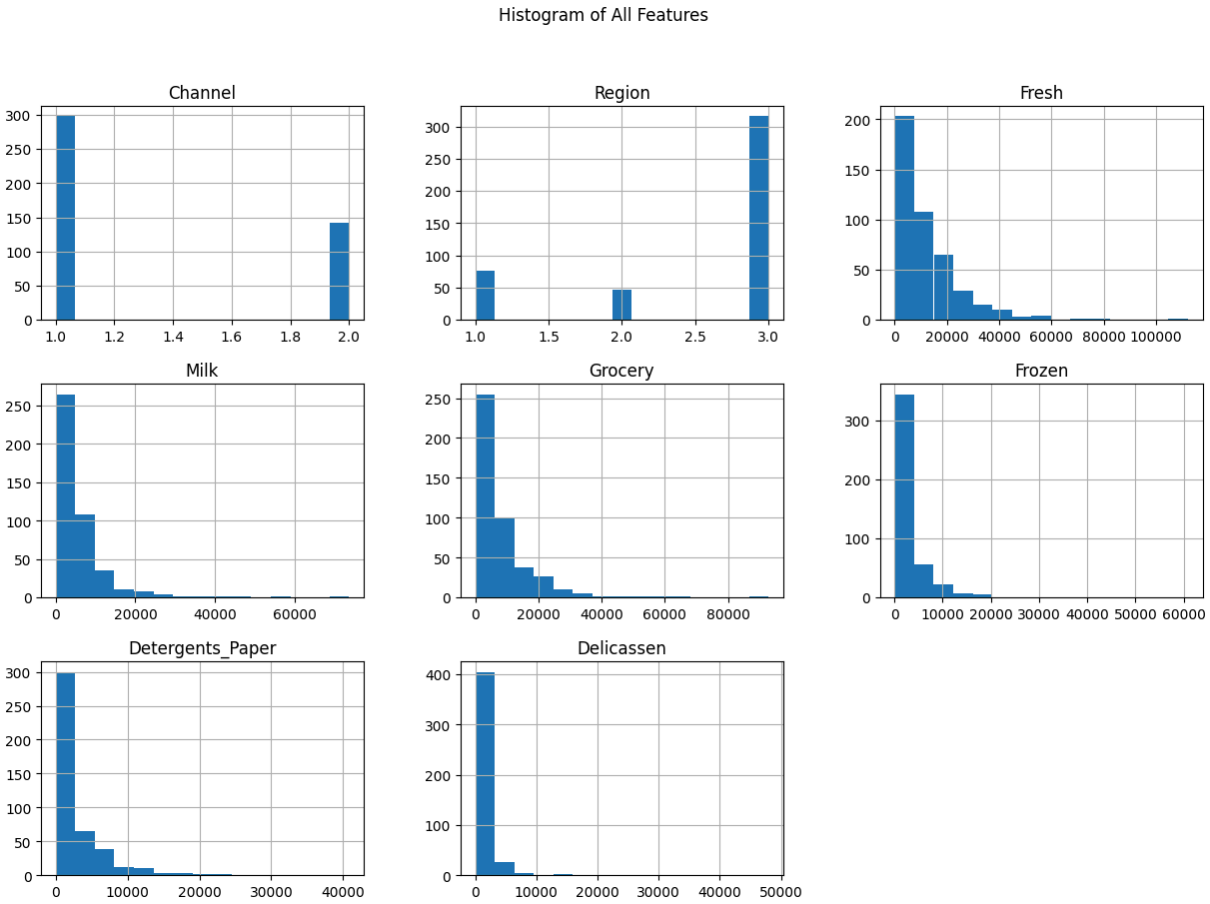
The dataset consists of the following features:

- **Channel:** The channel (Hotel/Restaurant/Café or Retail) where the customer belongs.
- **Region:** The geographic region of the customer.
- **Fresh:** Annual spending on fresh products.
- **Milk:** Annual spending on milk products.
- **Grocery:** Annual spending on grocery products.
- **Frozen:** Annual spending on frozen products.
- **Detergents_Paper:** Annual spending on detergents and paper products.
- **Delicassen:** Annual spending on delicatessen products.

4.2 Visualizing Data Distributions

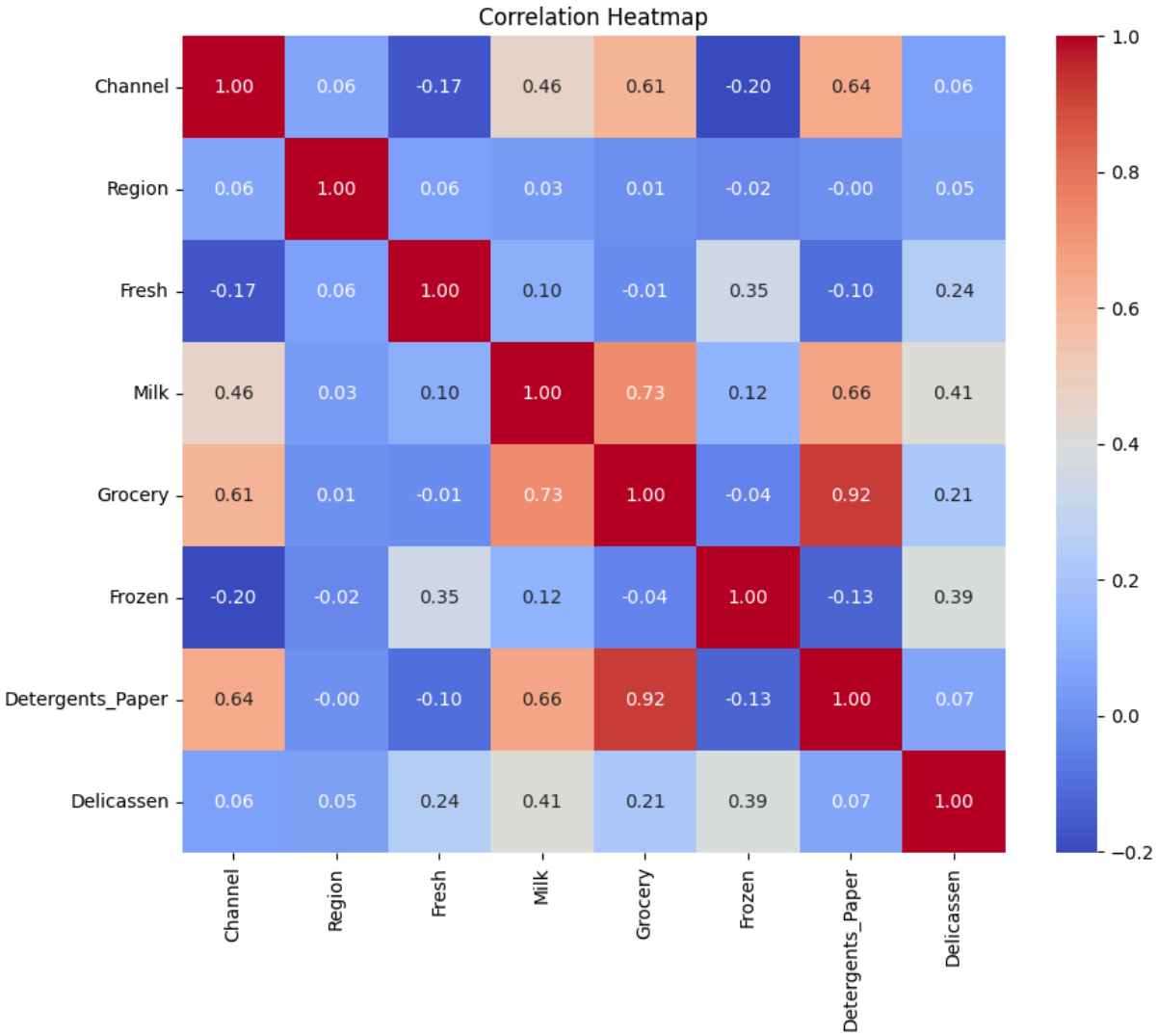
In [ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Box plots for each feature
plt.figure(figsize=(15, 10))
for i, column in enumerate(df.columns[2:], 1):
    plt.subplot(3, 2, i)
    sns.boxplot(y=df[column])
    plt.title(f'Distribution of {column}')
plt.tight_layout()
plt.show()

# Histograms for each feature
df.hist(figsize=(15, 10), bins=15)
plt.suptitle('Histogram of All Features')
plt.show()
```





4.3 Correlation Analysis

In [ ]:
```
# Correlation heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```



Observations:

- **Strong Correlations:** There is a strong correlation between `Grocery` and `Detergents_Paper`, indicating that customers who spend more on groceries also tend to spend more on detergents and paper products.
- **Weak Correlations:** Features like `Fresh` and `Delicassen` show weaker correlations with other features, suggesting they capture different aspects of customer behavior.

4.4 Data Transformation and Cleaning

Scaling the Data: Given the different scales of the features, we will apply Min-Max scaling to normalize the data before clustering.

In [ ]:
```
from sklearn.preprocessing import MinMaxScaler

# Apply Min-Max scaling
scaler = MinMaxScaler()
df_scaled = scaler.fit_transform(df.iloc[:, 2:])
```

Outliers:

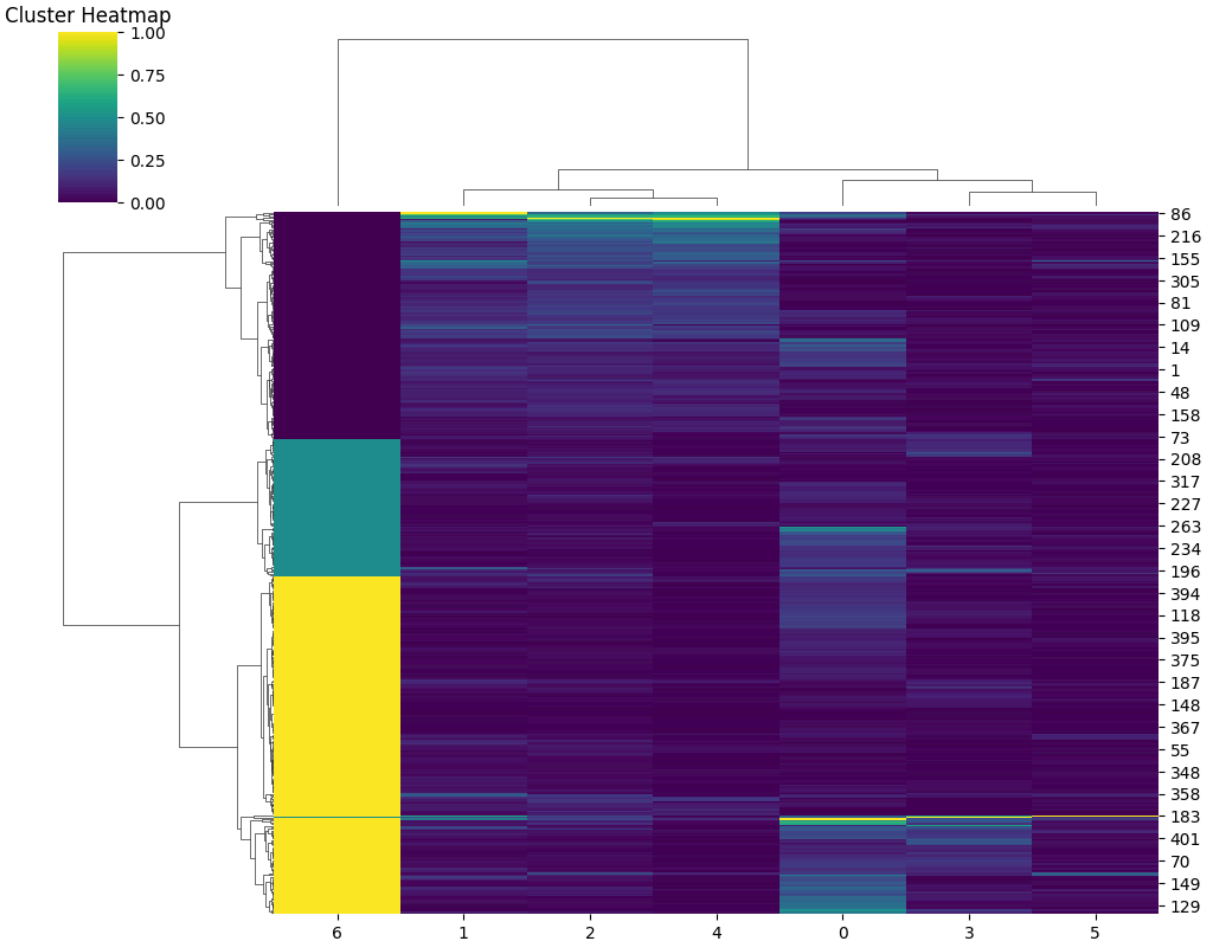
The box plots indicate the presence of outliers, particularly in `Fresh` and `Delicassen` expenditures. These will be handled by scaling but not removed, as they might represent important customer segments.

5. Perform Analysis Using Hierarchical Clustering

5.1 Clustering and Visualization

Cluster Heatmap:

In [ ]:
```
sns.clustermap(df_scaled, method='ward', metric='euclidean', cmap='viridis',
plt.title('Cluster Heatmap')
plt.show()
```



5.2 Reduce dimentionality by adapting PCA

In [ ]:
```
from sklearn.decomposition import PCA
# Reducing the dimensions of the data
pca = PCA(n_components = 2)
X_principal = pca.fit_transform(df_scaled)
X_principal = pd.DataFrame(X_principal)
```

```
X_principal.columns = ['P1', 'P2']

X_principal.head(2)
```

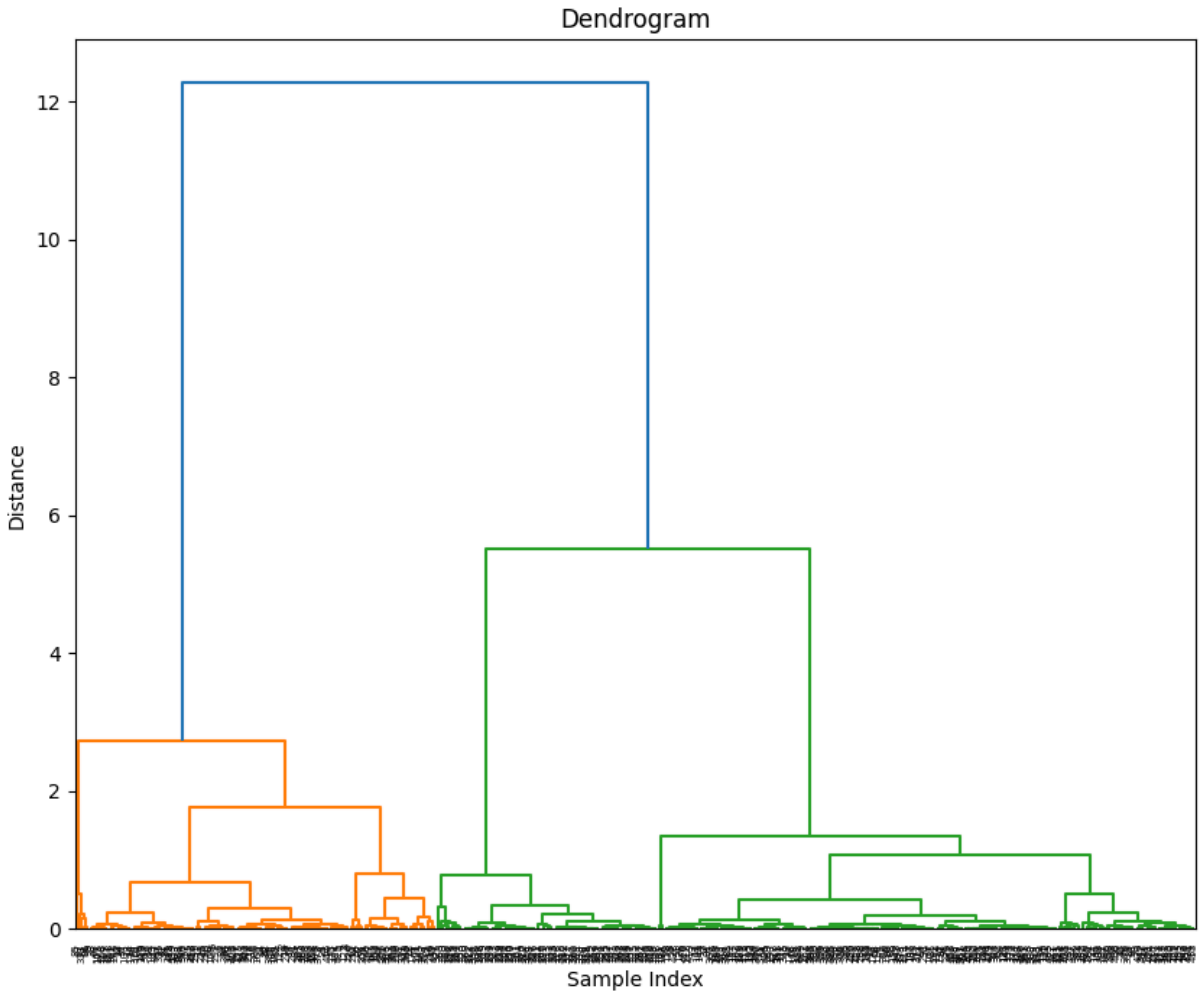
Out[ ]:

	P1	P2
0	-0.566285	-0.097817
1	-0.573399	-0.087729

5.3 Perform Hierarchical Clustering

In [ ]:

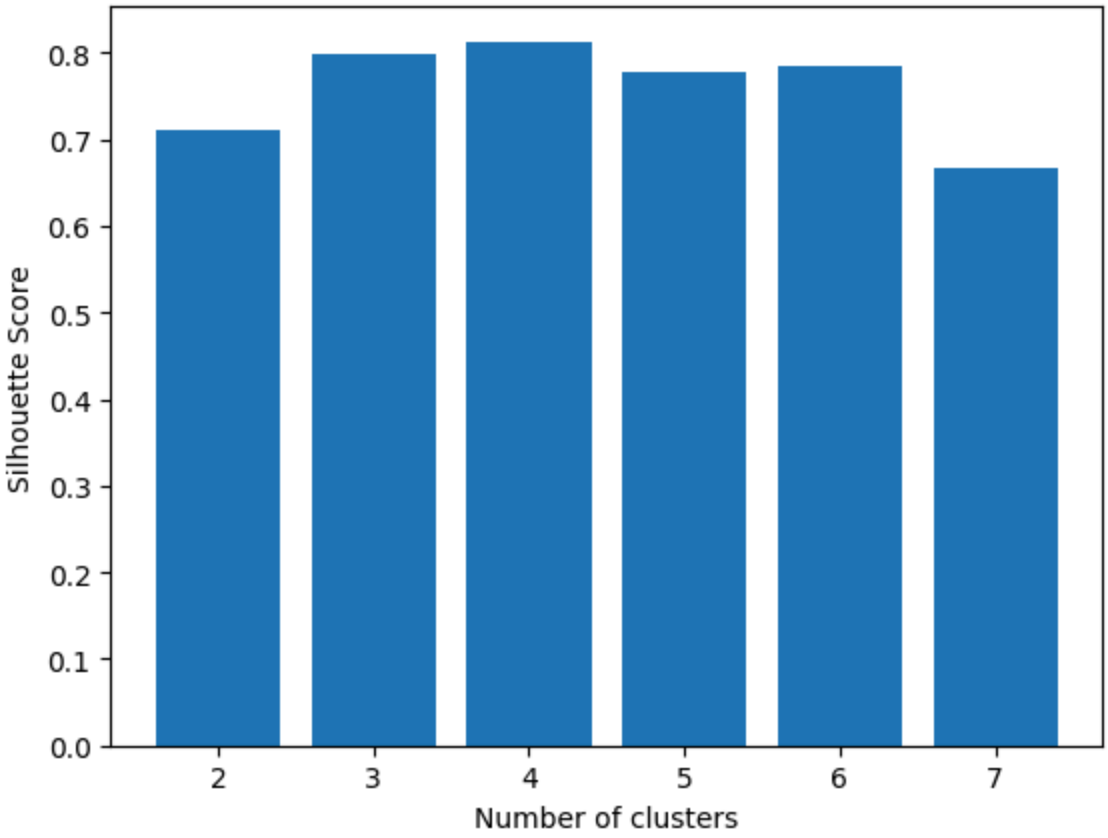
```
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.preprocessing import StandardScaler
Z = linkage(X_principal, method='ward')  # You can change 'ward' to 'complet


plt.figure(figsize=(10, 8))
dendrogram(Z)
plt.title('Dendrogram')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
plt.show()
```



5.4 Calculating clustering metrics

In [ ]:
```
silhouette_scores = []

for n_cluster in range(2, 8):

    silhouette_scores.append(silhouette_score(X_principal, AgglomerativeClus

print("silhouete score from 2 to 7 is", silhouette_scores)

# Plotting a bar graph to compare the results
k = [2, 3, 4, 5, 6,7]
plt.bar(k, silhouette_scores)
plt.xlabel('Number of clusters', fontsize = 10)
plt.ylabel('Silhouette Score', fontsize = 10)
plt.show()
```

```
silhouete score from 2 to 7 is [np.float64(0.7106606524591218), np.float64
(0.7978350734436583), np.float64(0.8131898806178876), np.float64(0.778588739
4088037), np.float64(0.7856795479118112), np.float64(0.6663371595031418)]
```



The silhouette score is a measure of how similar an object is to its own cluster compared to other clusters. The score ranges from -1 to 1, where a higher value indicates that the object is well-matched to its own cluster and poorly matched to neighboring clusters. Generally, a higher silhouette score indicates better-defined clusters.

Given the silhouette scores for different numbers of clusters (from 2 to 7):

- 2 clusters: 0.7106606524591218
- 3 clusters: 0.7978350734436583
- 4 clusters: 0.8131898806178876
- 5 clusters: 0.7785887394088037
- 6 clusters: 0.7856795479118112
- 7 clusters: 0.6663371595031418

Best Silhouette Score:

The highest silhouette score in this list is **0.8131898806178876**, which occurs when there are **4 clusters**.

Therefore, the best clustering configuration among these options is with 4 clusters, as it gives the highest silhouette score, indicating better-defined clusters.

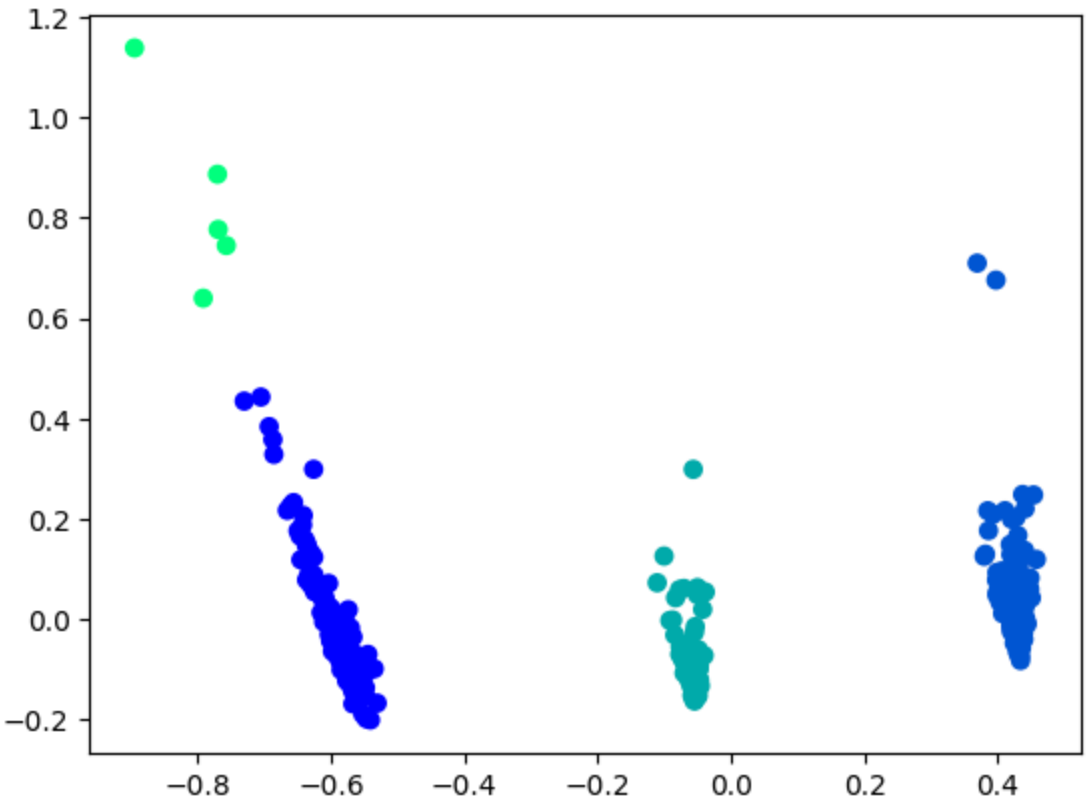
In [ ]:
```
agg = AgglomerativeClustering(n_clusters=4)
agg.fit(X_principal)
df['Cluster'] = agg.fit_predict(X_principal)
agg.labels_
```

Out[ ]:
```
array([0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1,
       1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0,
       0, 0, 0, 3, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 3, 0, 0, 1, 0,
       1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 3, 3, 1,
       1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 2, 0,
       2, 2, 0, 0, 2, 2, 2, 0, 2, 0, 2, 0, 2, 0, 2, 2, 0, 2, 0, 2, 0, 2,
       2, 2, 2, 0, 2, 2, 0, 2, 2, 2, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 0, 2, 2, 2, 2, 2, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       0, 2, 0, 2, 0, 2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 0, 2, 0, 2, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 2,
       2, 0, 2, 2, 0, 2, 2, 0, 2, 2, 2, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 0, 2, 3, 0, 0, 2, 2, 2, 2, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0,
       1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1,
       0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1])
```

In [ ]:
```
print(df['Cluster'].unique())
```

```
[0 1 3 2]
```

In [ ]:
```
plt.scatter(X_principal['P1'], X_principal['P2'],
            c = AgglomerativeClustering(n_clusters = 4).fit_predict(X_princip
plt.show()
```



In [ ]:
```
df_pca_merged = df.join(pd.DataFrame(X_principal, index=df.index))
cluster_summary = df.groupby('Cluster').agg(['mean', 'std'])
display(cluster_summary)
```

	Channel		Region		Fresh		
	mean	std	mean	std	mean	std	mean
Cluster							
0	2.0	0.0	2.605839	0.710816	8294.883212	8177.520805	9521.459854
1	1.0	0.0	3.000000	0.000000	13878.052133	14746.572913	3486.981043
2	1.0	0.0	1.321839	0.469890	12499.402299	11328.471783	3366.218391
3	2.0	0.0	2.800000	0.447214	25603.000000	14578.726059	43460.600000

This result presents the mean and standard deviation of different features (Channel, Region, Fresh, Milk, Grocery, Frozen, Detergents_Paper, and Delicassen) across four different clusters identified through hierarchical clustering. Let's break down and analyze the clusters:

Cluster 0:

- **Channel:** All customers in this cluster belong to Channel 2.
- **Region:** The customers are from the same region (Region 2).
- **Purchasing Behavior:**

- **Fresh:** The mean spending is approximately 8294 with a relatively high standard deviation of around 8177, indicating significant variation in spending on fresh products.
- **Milk, Grocery, Detergents_Paper, Delicassen:** Customers in this cluster have moderate to high spending across these categories. The standard deviations indicate varying levels of consistency in purchasing patterns across these categories.
- **Frozen:** The mean spending on Frozen products is 1616, with a standard deviation of 1757, indicating variability among the customers in this cluster.

Cluster 1:

- **Channel:** All customers in this cluster belong to Channel 1.
- **Region:** This cluster is uniform, with all customers from the same region (Region 3).
- **Purchasing Behavior:**
  - **Fresh:** Customers in this cluster spend heavily on fresh products, with a mean of 13878.
  - **Milk, Grocery, Detergents_Paper, Delicassen:** They have lower spending in these categories compared to Cluster 0.
  - **Frozen:** They spend similarly on frozen products as in Cluster 0, with relatively less variation.

Cluster 2:

- **Channel:** All customers are from Channel 1.
- **Region:** Similar to Cluster 1, but these customers are predominantly from Region 1.
- **Purchasing Behavior:**
  - **Fresh:** These customers spend moderately on fresh products (mean ~12499).
  - **Milk, Grocery, Detergents_Paper:** There is moderate spending in these categories.
  - **Frozen:** The spending on frozen products is higher than in Cluster 1 but still lower than Cluster 0.

Cluster 3:

- **Channel:** All customers belong to Channel 2.
- **Region:** Like Cluster 0, this cluster consists of customers from Region 2.
- **Purchasing Behavior:**
  - **Fresh:** The highest spending on fresh products is seen in this cluster, with a mean of 25603, which is significantly higher than other clusters.
  - **Milk, Grocery:** Extremely high spending in these categories (43460 for Milk and 61472 for Grocery), suggesting that this cluster likely consists of very large customers or institutions with bulk purchasing.

- **Detergents_Paper, Delicassen:** The spending is extremely high, with a significant variation, indicating a diverse group of customers or large institutions with significant consumption.
- **Frozen:** Spending on frozen products is moderately higher, indicating a broader range of product purchases.

Summary of Analysis:

- **Cluster 3** appears to consist of the most affluent or large customers with very high spending across all product categories. These could be institutions or large businesses.
- **Cluster 0** and **Cluster 1** represent moderate spenders, but Cluster 0 has more variation in spending.
- **Cluster 2** consists of moderate to low spenders with more consistency in their purchasing behavior.
- **Channels and Regions:** There's a clear distinction between Channels (1 and 2) and Regions (1, 2, 3) across clusters, indicating that customers from different channels and regions have distinctly different purchasing patterns.

Further Actions:

- **Targeting and Segmentation:** Marketing strategies can be tailored based on these clusters. For instance, Cluster 3 might be targeted with premium services and bulk discounts, while Cluster 2 might be offered more budget-friendly options.
- **Regional and Channel Insights:** Understanding the purchasing behavior of different regions and channels can inform strategic decisions like where to open new distribution centers or where to focus marketing efforts.

This analysis provides a strong foundation for business decisions, but further exploration (e.g., using other clustering techniques or integrating more data) could provide even deeper insights.