

KELOMPOK 6

IMPLEMENTASI ALGORITMA KLASSTERING DENSITY-BASED CLUSTERING NON-PARAMETRIC ALGORITHM (DBSCAN)



Disusun Oleh:

Rendi Wijaya (D121211020)

Andi Muhammad Fauzan Agung (D121211094)

Muh. Rhayyan Zhakhi (D121211095)

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

MAKASSAR

2024

A. PENDAHULUAN

1. Problem Statement

Analisis perilaku pelanggan merupakan analisis rinci mengenai pelanggan ideal sebuah perusahaan. Dimana analisis ini dapat membantu perusahaan untuk memahami bagaimana kebiasaan dan perilaku pelanggan sehingga dapat memenuhi kebutuhan dari berbagai pelanggan. Pada tugas kali ini kami mengimplementasikan algoritma clustering DBSCAN pada sample data customer personality yang kami dapatkan melalui platform Kaggle. Dimana sample data berupa berupa atribut-atribut yang akan memiliki korelasi terhadap pelanggan.

Source : [Customer-personality-analysis](#)

2. Project Objectives

- Menentukan bagaimana karakteristik dan perilaku pelanggan
- Mengelompokkan pelanggan yang serupa berdasarkan karakteristik dan perilaku
- Membuat model prediktif untuk memprediksi pelanggan yang akan merespon sebuah kampanye perusahaan

3. Sample Data

Berikut adalah bentuk sampel data yang akan kami analisis:

ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProducts	MntSweetProducts	MntGoldProds
5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	58	635	88	546	172	88	88
2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	38	11	1	6	2	1	6
4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	26	426	49	127	111	21	42
6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	26	11	4	20	10	3	5
5324	1981	PhD	Married	58293.0	1	0	19-01-2014	94	173	43	118	46	27	15

Keterangan Atribut:

1) People

- ID: Customer's unique identifier
- Year_Birth: Customer's birth year
- Education: Customer's education level
- Marital_Status: Customer's marital status
- Income: Customer's yearly household income
- Kidhome: Number of children in customer's household
- Teenhome: Number of teenagers in customer's household
- Dt_Customer: Date of customer's enrollment with the company
- Recency: Number of days since customer's last purchase
- Complain: 1 if the customer complained in the last 2 years, 0 otherwise

2) Products

- MntWines: Amount spent on wine in last 2 years
- MntFruits: Amount spent on fruits in last 2 years
- MntMeatProducts: Amount spent on meat in last 2 years
- MntFishProducts: Amount spent on fish in last 2 years
- MntSweetProducts: Amount spent on sweets in last 2 years

- MntGoldProds: Amount spent on gold in last 2 years
-

3) Promotion

- NumDealsPurchases: Number of purchases made with a discount
- AcceptedCmp1: 1 if customer accepted the offer in the 1st campaign, 0 otherwise
- AcceptedCmp2: 1 if customer accepted the offer in the 2nd campaign, 0 otherwise
- AcceptedCmp3: 1 if customer accepted the offer in the 3rd campaign, 0 otherwise
- AcceptedCmp4: 1 if customer accepted the offer in the 4th campaign, 0 otherwise
- AcceptedCmp5: 1 if customer accepted the offer in the 5th campaign, 0 otherwise
- Response: 1 if customer accepted the offer in the last campaign, 0 otherwise

4) Place

- NumWebPurchases: Number of purchases made through the company's website
- NumCatalogPurchases: Number of purchases made using a catalogue
- NumStorePurchases: Number of purchases made directly in stores
- NumWebVisitsMonth: Number of visits to company's website in the last month

B. PRE-PROCESSING

1. Features Engineering (Modifikasi Kolom)

```

1 #Feature Engineering
2
3 #Create 'Age' feature from customer's birth year
4 customer_data['Age'] = customer_data.Year_Birth.apply(Lambda x: 2021 - int(x))
5
6 #Create 'Days_Since_Customer' feature from time the customer enrolled
7 customer_data['Dt_Customer'] = pd.to_datetime(customer_data.Dt_Customer)
8 now = datetime.now()
9 customer_data['Days_Since_Customer'] = customer_data.Dt_Customer.apply(Lambda x: (now - x).total_seconds() / (60 * 60 * 24))
10
11 #Create 'Fam_Size' feature from the marriage status, number of kids/teens
12 marital_map = {'Absurd': 1, 'Alone': 1, 'YOLO': 1, 'Single': 1,
13               'Married': 2, 'Together': 2, 'Widow': 1, 'Divorced': 1}
14 customer_data['Marital_Status'] = customer_data.Marital_Status.map(marital_map) #Maps all singles as 1, couples as 2
15 customer_data['Num_Kids'] = customer_data.Kidhome.values + customer_data.Teenhome.values
16 customer_data['Fam_Size'] = customer_data.Marital_Status.values + customer_data.Num_Kids.values
17
18 #Create 'Num_Accepted' feature from the sum of previous marketing campaigns that were accepted by the customer
19 customer_data['Num_Accepted'] = customer_data.AcceptedCmp1.values + customer_data.AcceptedCmp2.values + \
20                               customer_data.AcceptedCmp3.values + customer_data.AcceptedCmp4.values + \
21                               customer_data.AcceptedCmp5.values
22
23 #Create 'MntTotal' for total amount spent on all items
24 customer_data['MntTotal'] = customer_data['MntWines'].values + customer_data['MntFruits'].values + \
25                             customer_data['MntMeatProducts'].values + customer_data['MntFishProducts'].values + \
26                             customer_data['MntWines'].values + customer_data['MntSweetProducts'].values + \
27                             customer_data['MntGoldProds'].values
28
29 #Drops the unnecessary features from the original dataset
30 customer_data.drop(['Dt_Customer', 'Year_Birth', 'AcceptedCmp1', 'AcceptedCmp2',
31                   'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'Kidhome', 'Teenhome',
32                   'Z_CostContact', 'Z_Revenue', 'Num_Kids', 'Marital_Status'],
33                  axis=1, inplace=True)
34 customer_data.head()

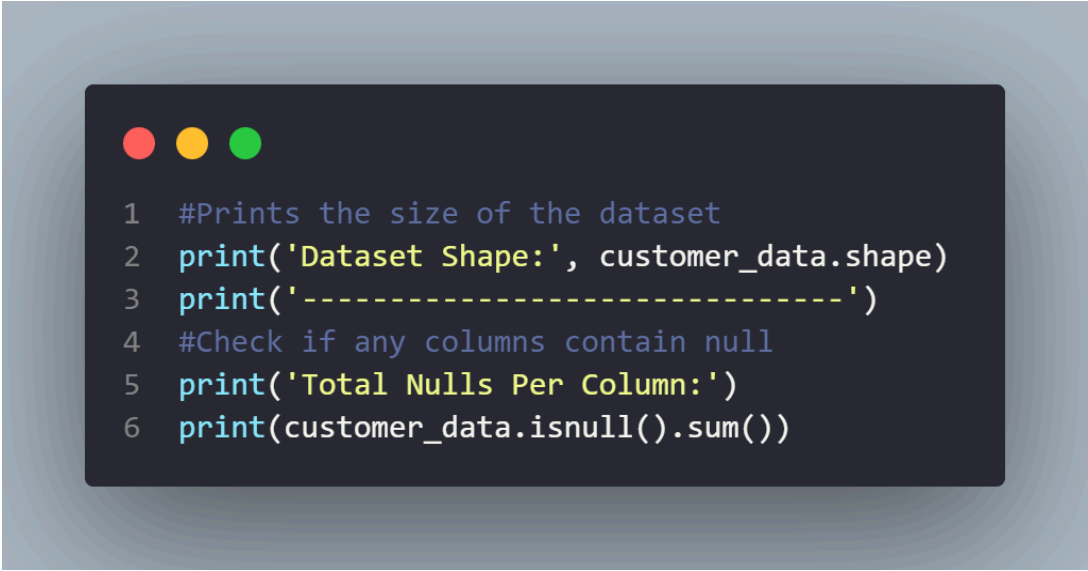
```

Kode tersebut bertujuan untuk melakukan beberapa modifikasi pada features (atribut) pada dataset. Beberapa features yang dimodifikasi yaitu dengan membuat kolom baru berupa Age, Days_Since_Customer, Fam_Size, Num, Accepted, MntTotal. Dan juga dilakukan modifikasi

seperti menghapus kolom-kolom yang tidak diperlukan dari dataset setelah proses pembuatan kolom-kolom (atribut) baru sebelumnya. Dengan melakukan feature engineering ini maka kita dapat mengambil informasi yang relevan saja dari dataset.

2. Missing Value (Nilai Kosong)

Setelah melakukan modifikasi kolom pada tahap features engineering kita dapat melakukan proses untuk mencari apakah terdapat missing value atau nilai yang kosong pada dataset. Jika terdapat missing value maka kita perlu untuk melakukan modifikasi dengan mengisi missing value tersebut.



```
1 #Prints the size of the dataset
2 print('Dataset Shape:', customer_data.shape)
3 print('-----')
4 #Check if any columns contain null
5 print('Total Nulls Per Column:')
6 print(customer_data.isnull().sum())
```

```
Dataset Shape: (2240, 21)
```

```
-----  
Total Nulls Per Column:
```

Education	0
Income	24
Recency	0
MntWines	0
MntFruits	0
MntMeatProducts	0
MntFishProducts	0
MntSweetProducts	0
MntGoldProds	0
NumDealsPurchases	0
NumWebPurchases	0
NumCatalogPurchases	0
NumStorePurchases	0
NumWebVisitsMonth	0
Complain	0
Response	0
Age	0
Days_Since_Customer	0
Fam_Size	0
Num_Accepted	0
MntTotal	0

```
dtype: int64
```

Setelah memeriksa missing value pada dataset kita menemukan bahwa terdapat nilai kosong pada kolom “income” sebanyak 24. Sehingga kita perlu untuk mengisi nilai kosong tersebut sebelum masuk ke tahap selanjutnya. Berikut adalah kode untuk mengisi nilai kosong tersebut dengan menggunakan nilai rata-rata dari kolom “income”.

```
1 #Imputes the mean  
2 imputer = SimpleImputer(strategy='mean')  
3 imputer.fit(customer_data.Income.values.reshape(-1,1))  
4 customer_data['Income'] = imputer.transform(customer_data.Income.values.reshape(-1,1))
```

3. Encoded

Tahap selanjutnya setelah memastikan bahwa sampel data tidak memiliki missing value atau nilai kosong maka kita bisa melakukan proses dimensi reduksi. Akan tetapi sebelum melakukan proses dimensi reduksi kita masih perlu untuk melakukan proses modifikasi terhadap kolom yang memiliki value kategori menjadi numerik.

```

1 #Remove the 'Response' column because it is the target of future predictive model
2 X, y = customer_data.drop('Response', axis=1).values, customer_data['Response'].values
3
4 #Creates a column transformer that sends 'Education' to be encoded and rest scaled
5 ct = ColumnTransformer([
6     ('catagoric', OneHotEncoder(), [0]),
7     ('numeric', StandardScaler(), list(range(1, len(X.T))))
8 ])
9
10 #Sends the data through the column transformer
11 X_transformed = ct.fit_transform(X)
12 print('Preprocessed Data:')
13 print(X_transformed[0])

```

C. DBSCAN CLUSTERING

1. Dimensi Reduksi (PCA)

Dimensi reduksi adalah proses mengurangi jumlah fitur (dimensi) dalam dataset, tetapi tetap mempertahankan sebanyak mungkin informasi yang relevan. Hal ini berguna ketika dataset memiliki terlalu banyak fitur, yang dapat menyebabkan masalah seperti overfitting, meningkatkan kompleksitas model, dan memperlambat waktu komputasi.

PCA adalah teknik yang paling umum digunakan untuk reduksi dimensi. PCA mencari kombinasi linear baru dari fitur yang ada sehingga variabilitas data dijelaskan dengan sedikit dimensi yang mungkin. Dengan memilih hanya sejumlah komponen utama yang paling signifikan, dimensi data dapat dikurangi secara signifikan.

```

1 #Create instance of Principal Component Analysis in order to reduce dimensionality while maintaining variance
2 #n_components=3 will be chosen in order to visualize the data better
3 pca = PCA(n_components=3)
4
5 #fit to dataset
6 pca.fit(X_transformed)
7
8 #create dimentionality reduced dataset
9 X_reduced = pca.transform(X_transformed)
10
11 print('Dimentionality Reduced Data:')
12 print(X_reduced[0])


```

```

Dimentionality Reduced Data:
[ 4.1962933  1.01624418 -2.28447375]

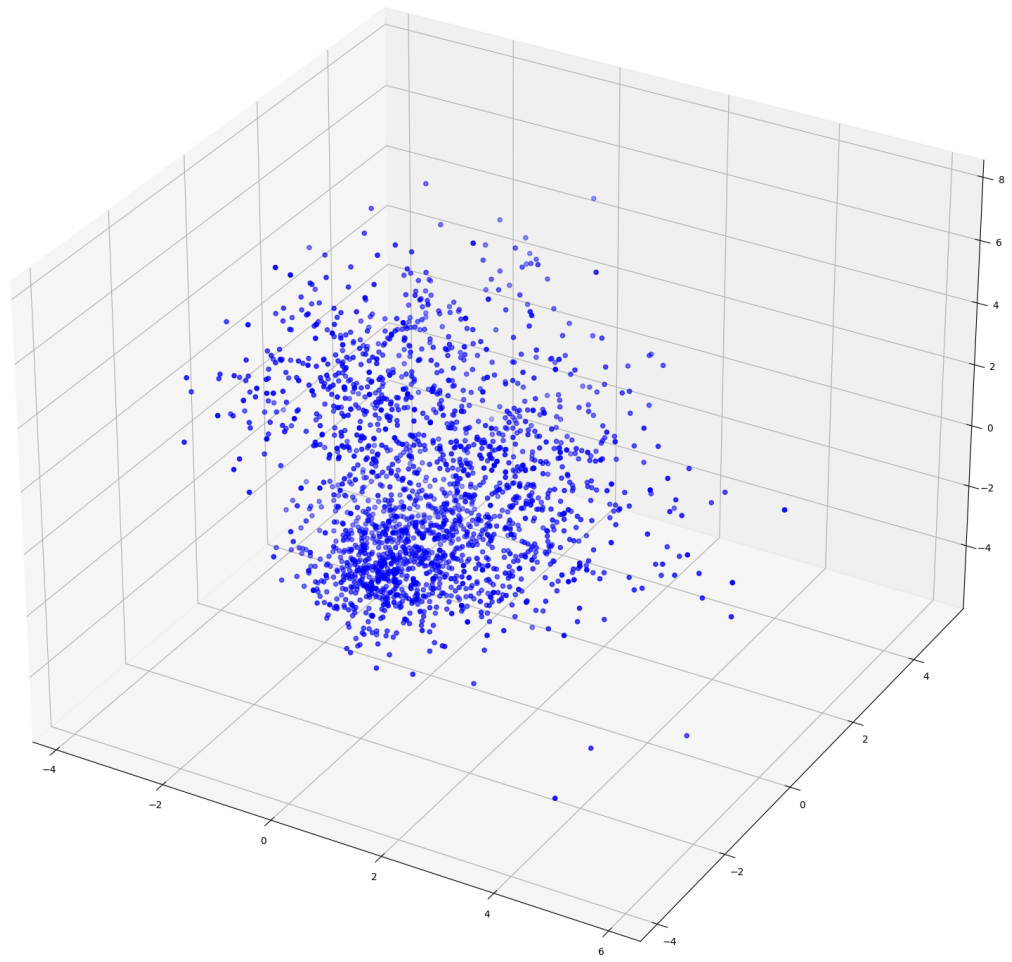
```

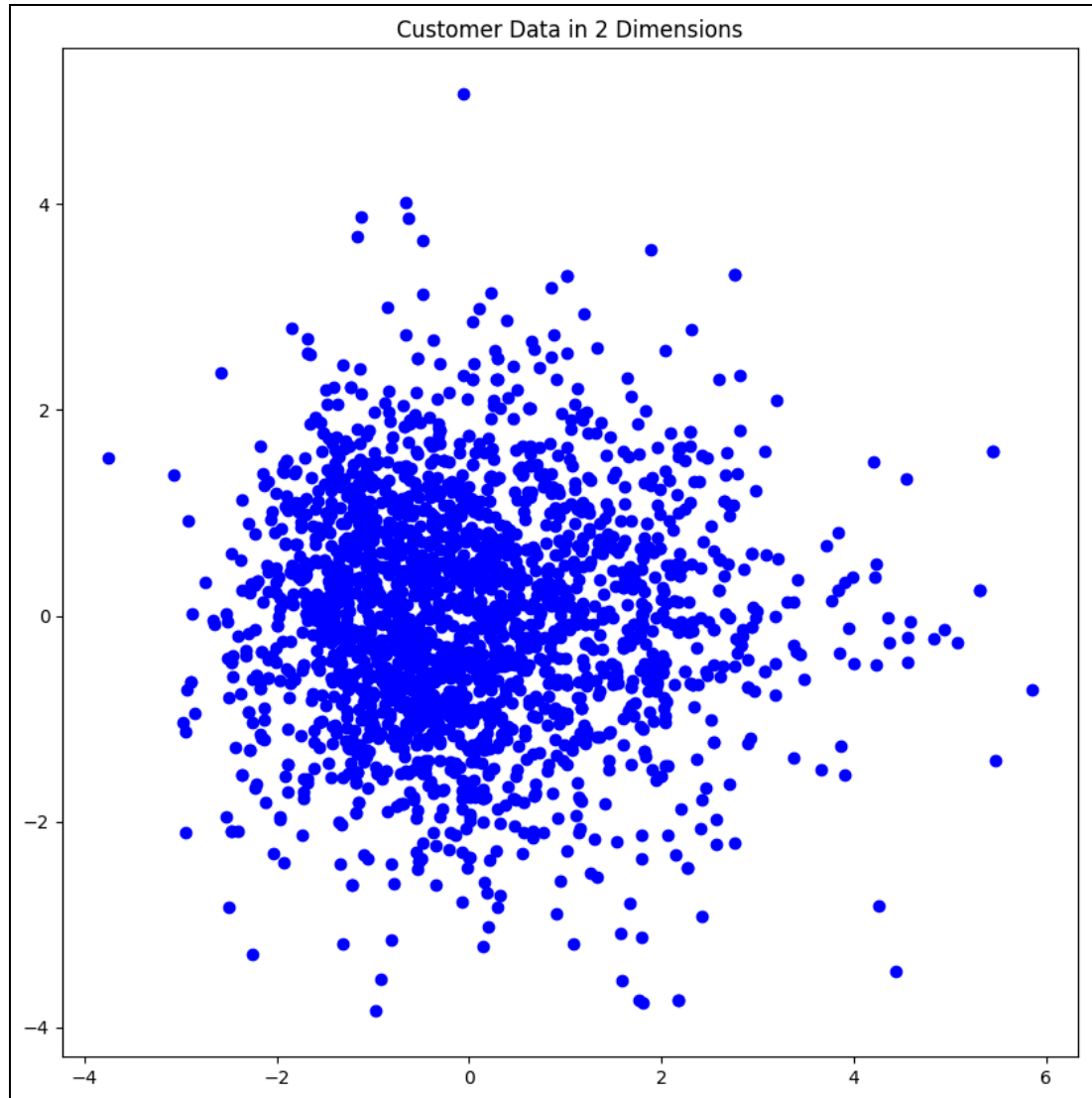
2. Visualisasi Data



```
1 #plot the 3d dataset
2 fig = plt.figure(figsize=(20,20))
3 ax = fig.add_subplot(111, projection="3d")
4 ax.scatter(X_reduced.T[1],X_reduced.T[2],X_reduced.T[0], c="blue")
5 ax.set_title("Customer Data in 3 Dimensions")
6 plt.show()
7
8 #plot the 2d dataset
9 plt.figure(figsize=(10, 10))
10 plt.scatter(X_reduced.T[1], X_reduced.T[2], c="blue")
11 plt.title('Customer Data in 2 Dimensions')
12 plt.xlabel('')
13 plt.ylabel('')
14 plt.show()
```

Customer Data in 3 Dimensions





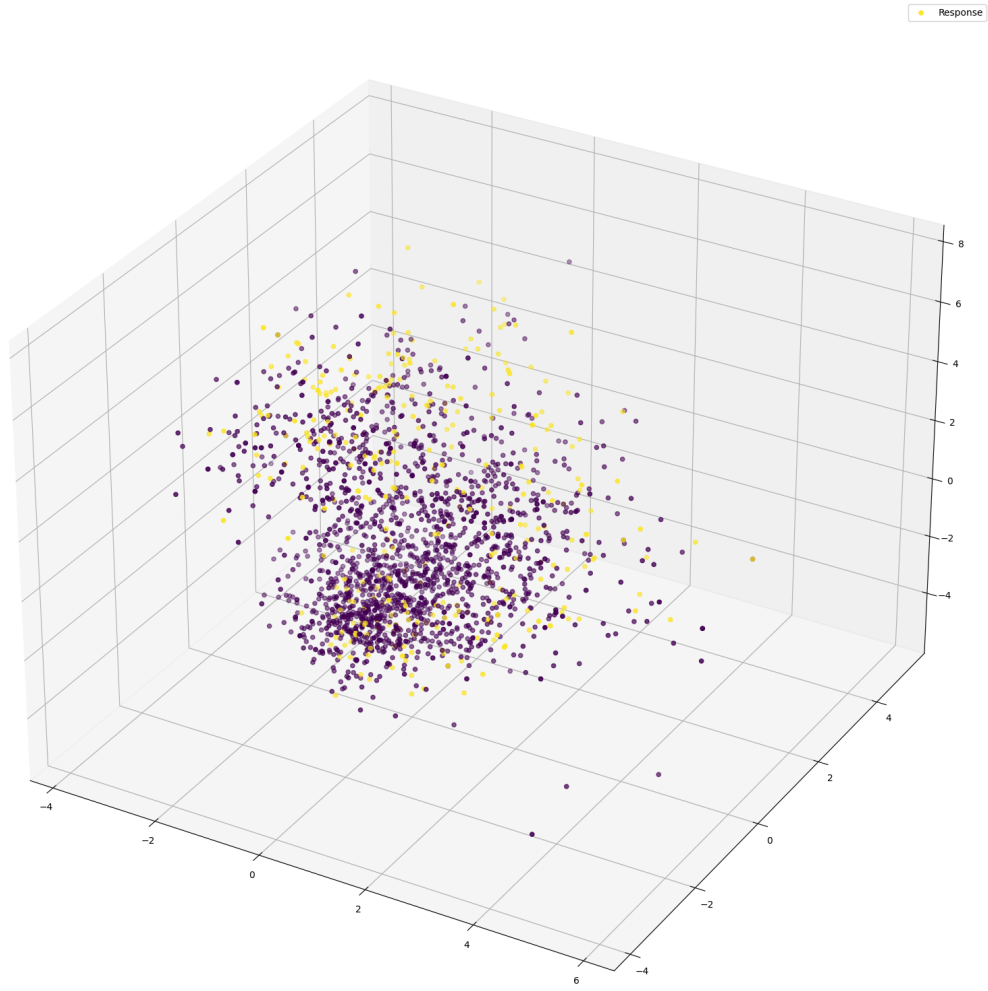
Kode tersebut menghasilkan sebuah plot 3D dan 2D dari dataset yang telah direduksi dimensinya sebelumnya. Pada plot tersebut, sumbu x mewakili nilai dari dimensi kedua ($X_reduced.T[1]$), sumbu y mewakili nilai dari dimensi ketiga ($X_reduced.T[2]$), dan sumbu z mewakili nilai dari dimensi pertama ($X_reduced.T[0]$). Titik-titik pada plot diwarnai biru dan mewakili data pelanggan dalam tiga dimensi. Judul plot adalah "Customer Data in 3 Dimensions".

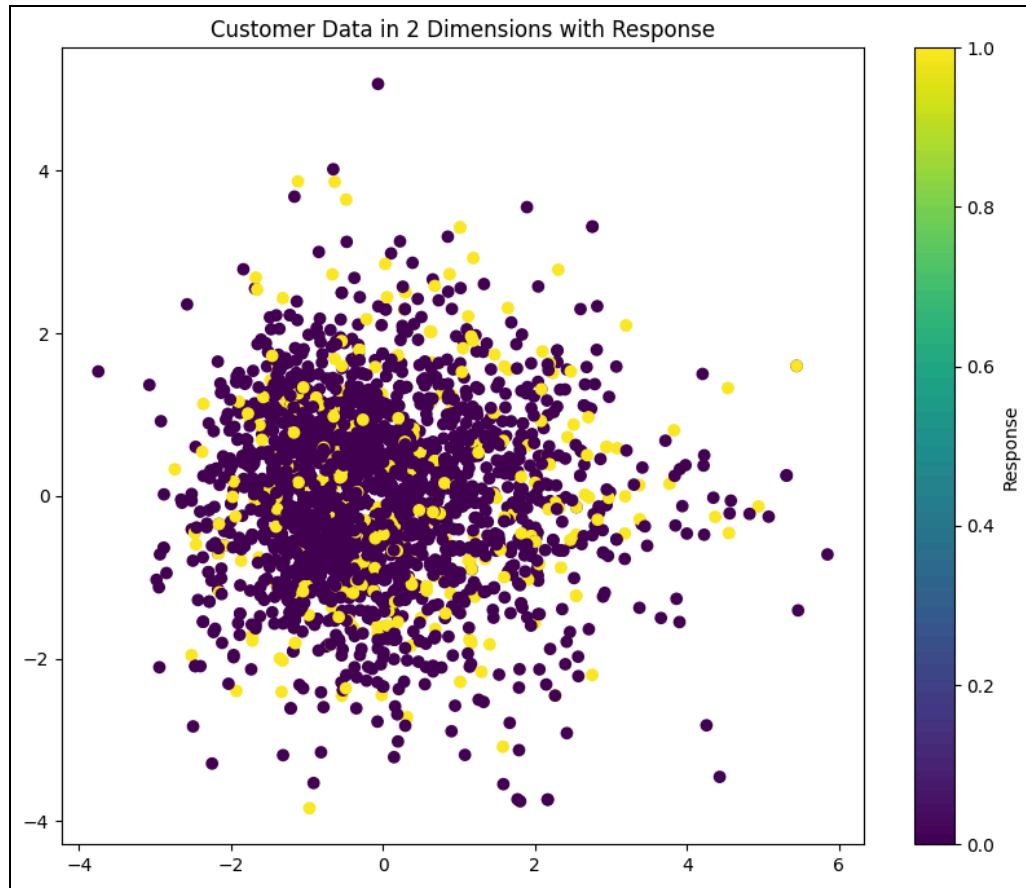
```
1 # plot the 3d dataset
2 fig = plt.figure(figsize=(20,20))
3 ax = fig.add_subplot(111, projection="3d")
4 ax.scatter(X_reduced.T[1],X_reduced.T[2],X_reduced.T[0], c=y)
5 ax.set_title("Customer Data in 3 Dimensions with Response")
6 ax.legend(['Response'])
7 plt.show()
8
9 #plot the 2d dataset
10 plt.figure(figsize=(10, 8))
11 plt.scatter(X_reduced.T[1], X_reduced.T[2], c=y, cmap='viridis')
12 plt.title("Customer Data in 2 Dimensions with Response")
13 plt.xlabel("")
14 plt.ylabel("")
15 plt.colorbar(Label='Response')
16 plt.show()
17
```

Kode di atas akan menghasilkan plot 3D dan 2D dari dataset, dengan nilai pada sumbu x diambil dari dimensi kedua, nilai pada sumbu y diambil dari dimensi ketiga, dan nilai pada sumbu z diambil dari dimensi pertama dari dataset yang telah direduksi (X_reduced). Warna titik-titik pada plot akan ditentukan berdasarkan nilai dari variabel y.

Selanjutnya kita akan mencoba melihat bagaimana bentuk visualisasi plot pada customer data sebelumnya dengan menggunakan feature respons untuk mengetahui bagaimana korelasi antara customer data dengan respons

Customer Data in 3 Dimensions with Response





3. Clustering Customer

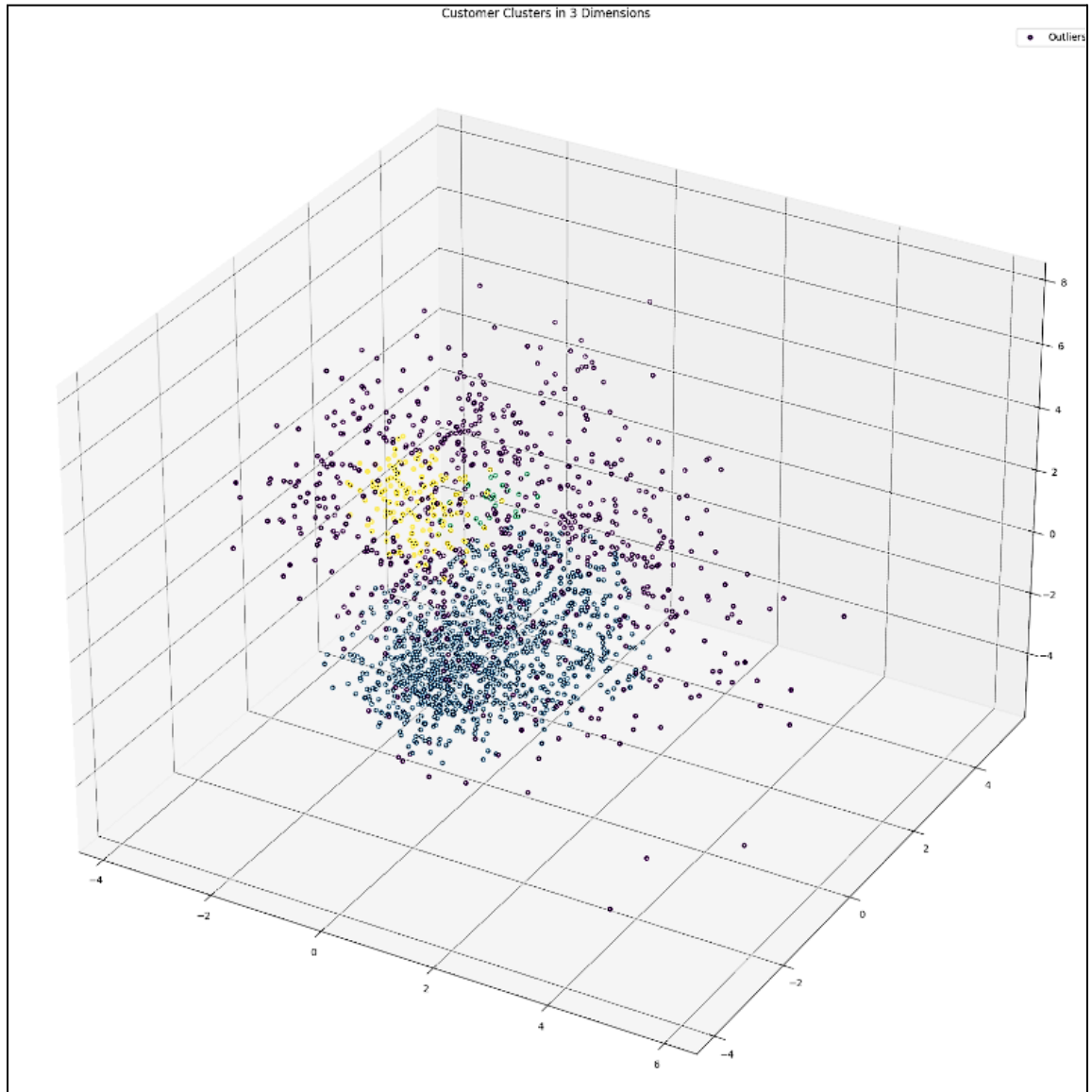
```
1 #Create an instance of DBSCAN to create non spherical clusters based on data density
2 db = DBSCAN(eps=0.726, min_samples=26)
3
4 #fit to the dimensionality reduced dataset
5 db.fit(X_reduced)
6
7 #identify the clusters
8 clusters = db.labels_
9
10 #display metrics/sample
11 n_clusters_ = len(set(clusters)) - (1 if -1 in clusters else 0)
12 n_noise_ = list(clusters).count(-1)
13
14 print('Cluster Predictions')
15 print('-----')
16 print("Number of clusters: %d" % n_clusters_)
17 print("Number of noise points: %d" % n_noise_)
18 print('Number of points per cluster:')
19 for i in range(n_clusters_):
20     print('Cluster', i, ':', len(clusters[clusters==i]))
```

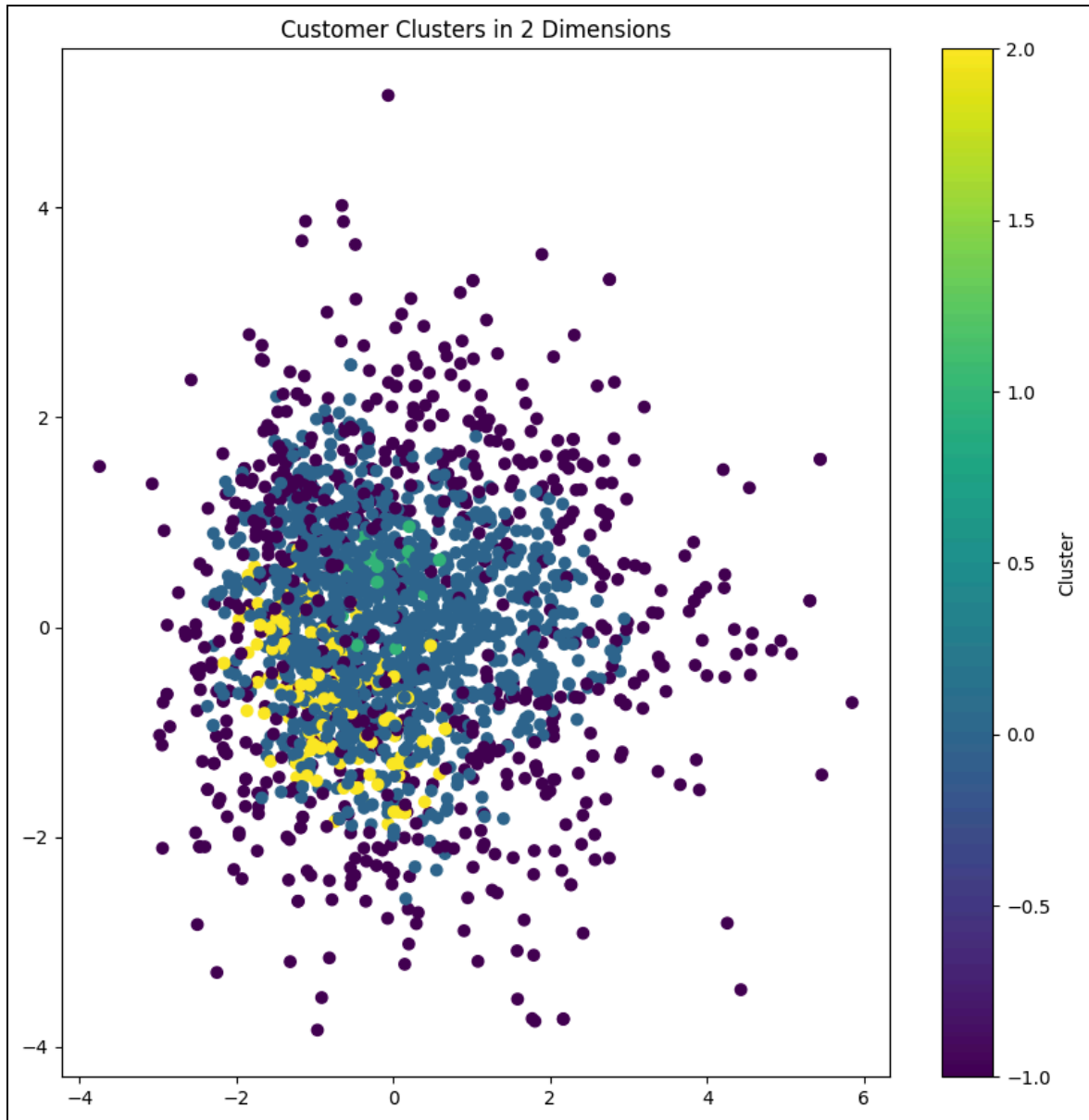
Selanjutnya kita akan mencoba untuk mengimplementasikan algoritma DBSCAN pada sampel data yang telah direduksi menggunakan PCA sebelumnya. Pada kode tersebut kita juga akan mengidentifikasi cluster-cluster dalam data setelah dimensi direduksi. Pada algoritma DBSCAN ini kita menggunakan eps dengan nilai 0.726 dengan nilai min_samples sebesar 26.

```
Cluster Predictions
-----
Number of clusters: 3
Number of noise points: 753
Number of points per cluster:
Cluster 0 : 1285
Cluster 1 : 34
Cluster 2 : 168
```

Dengan menggunakan algoritma DBSCAN pada sampel data yang telah direduksi kita dapat mengetahui bahwa terdapat 3 cluster customer yang dihasilkan algoritma DBSCAN dengan noise sekitar 750. Cluster 0 : 1285, Cluster 1: 34, Cluster 2: 168

D. VISUALISASI HASIL KLASTER





Keterangan Plot :

- Ungu : Outliers dari customer
- Biru : Cluster customer (0)
- Hijau : Cluster customer (1)
- Kuning : Cluster customer (2)

```

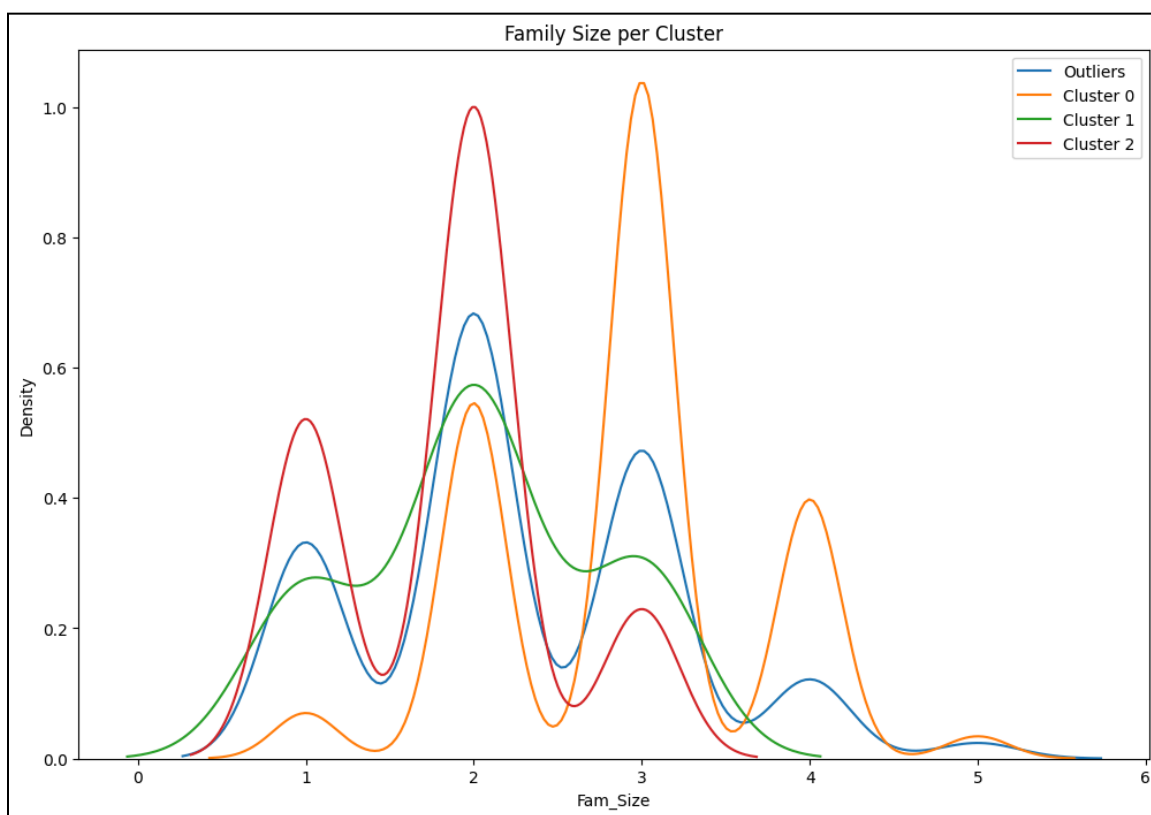
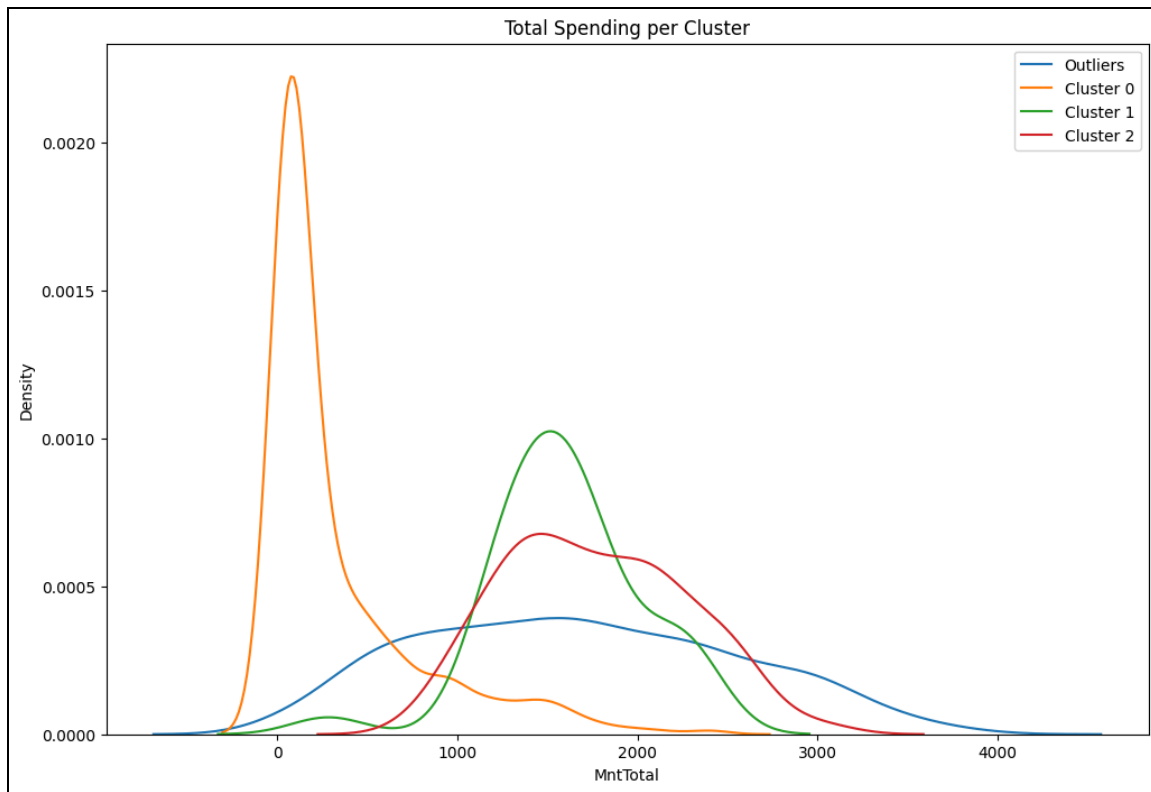
1 #append the clusters to the original dataset
2 customer_data['Cluster'] = clusters
3
4 #group by cluster and calculate how many responses there were per cluster
5 cluster_grp = customer_data.groupby('Cluster').Response.agg(['sum', 'count'])
6 cluster_grp['percent_resp'] = cluster_grp['sum'].values / cluster_grp['count'].values
7 cluster_grp

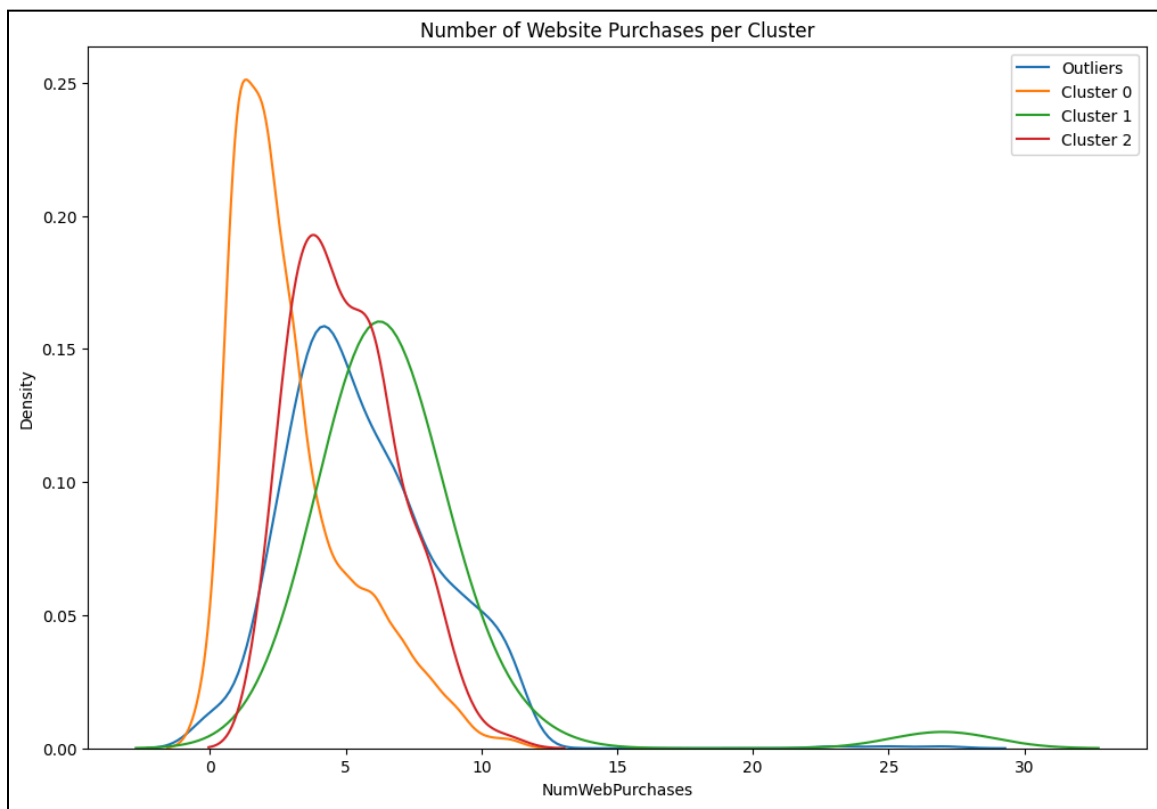
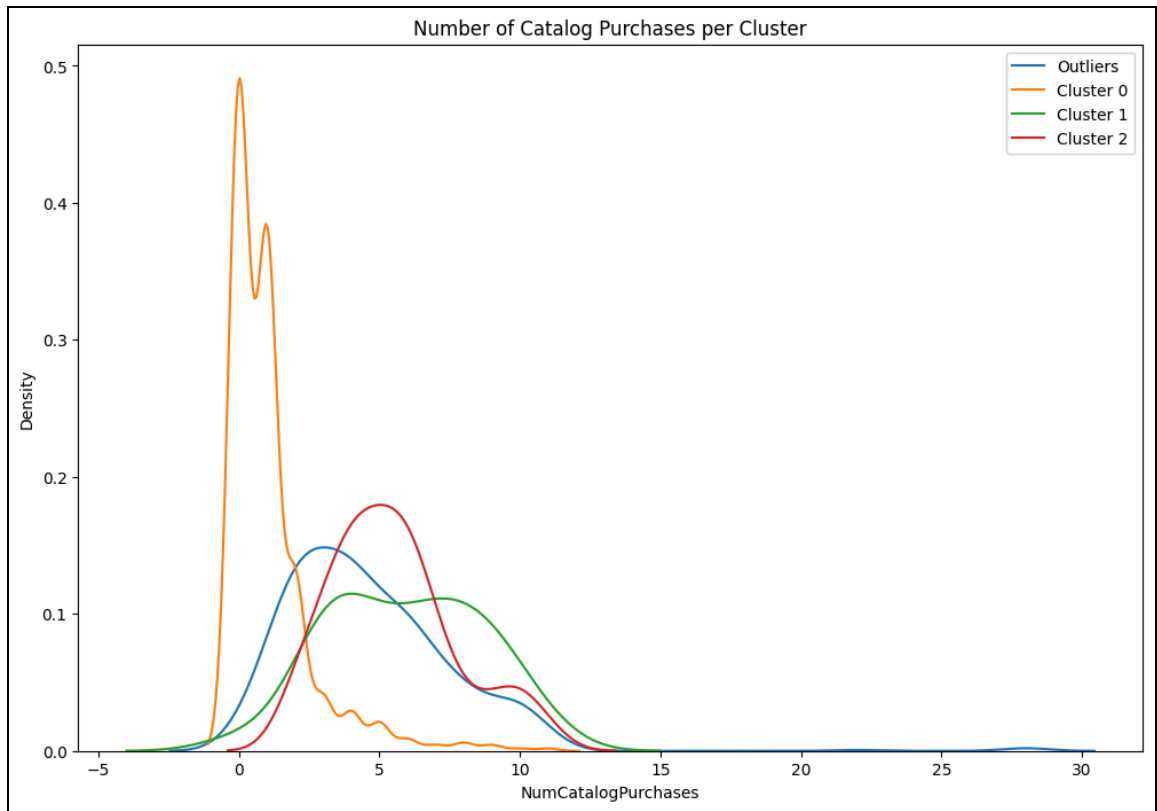
```

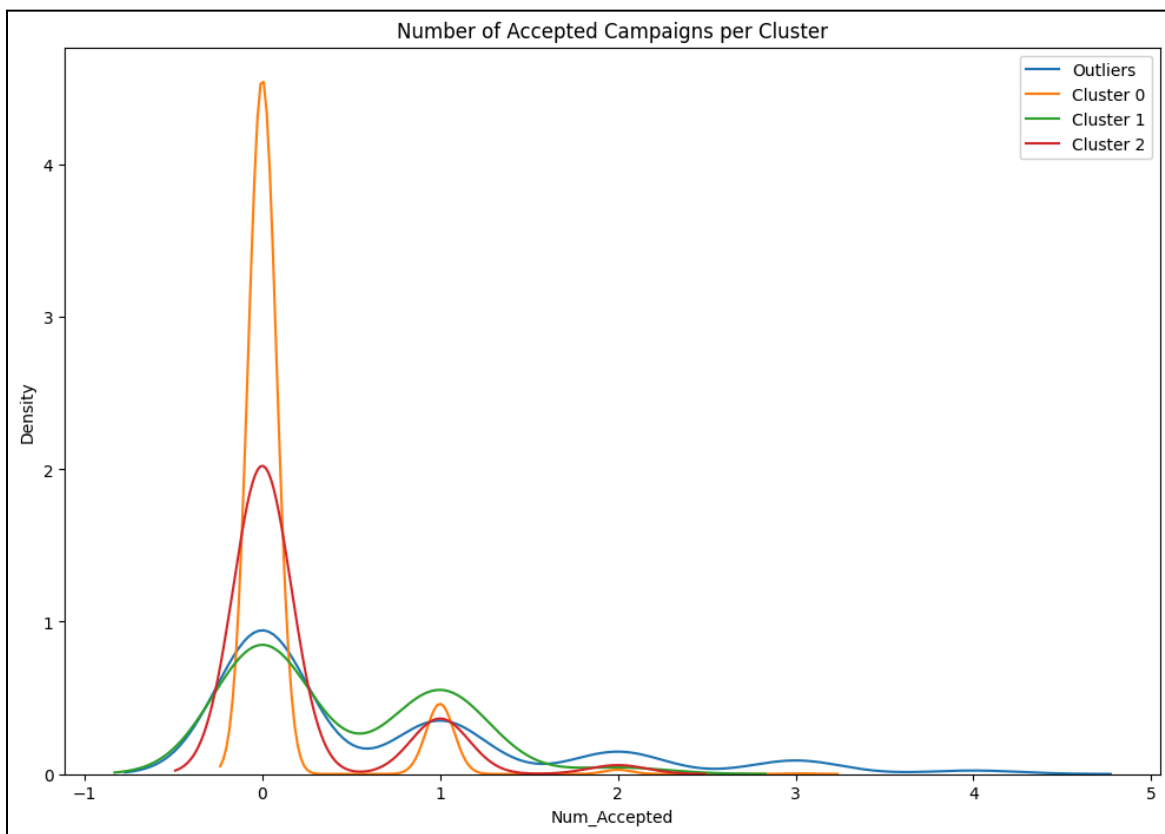
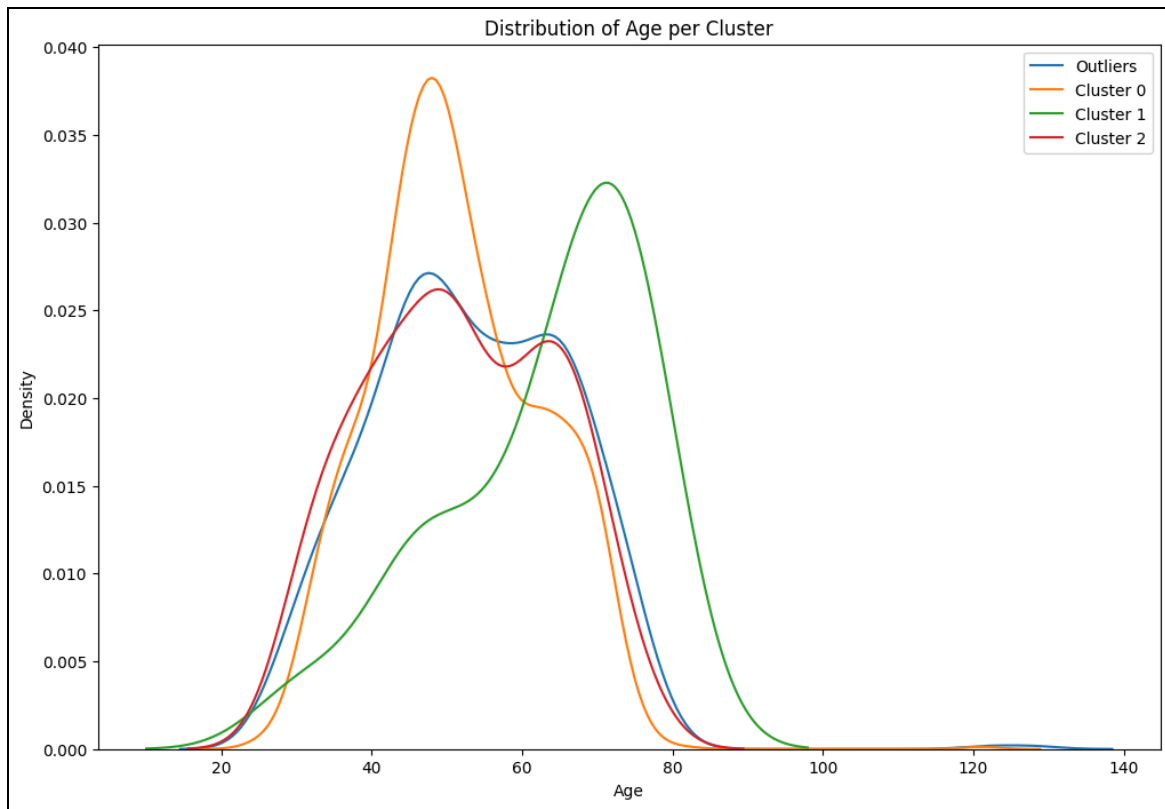
	sum	count	percent_resp
Cluster			
-1	177	753	0.235060
0	118	1285	0.091829
1	6	34	0.176471
2	33	168	0.196429

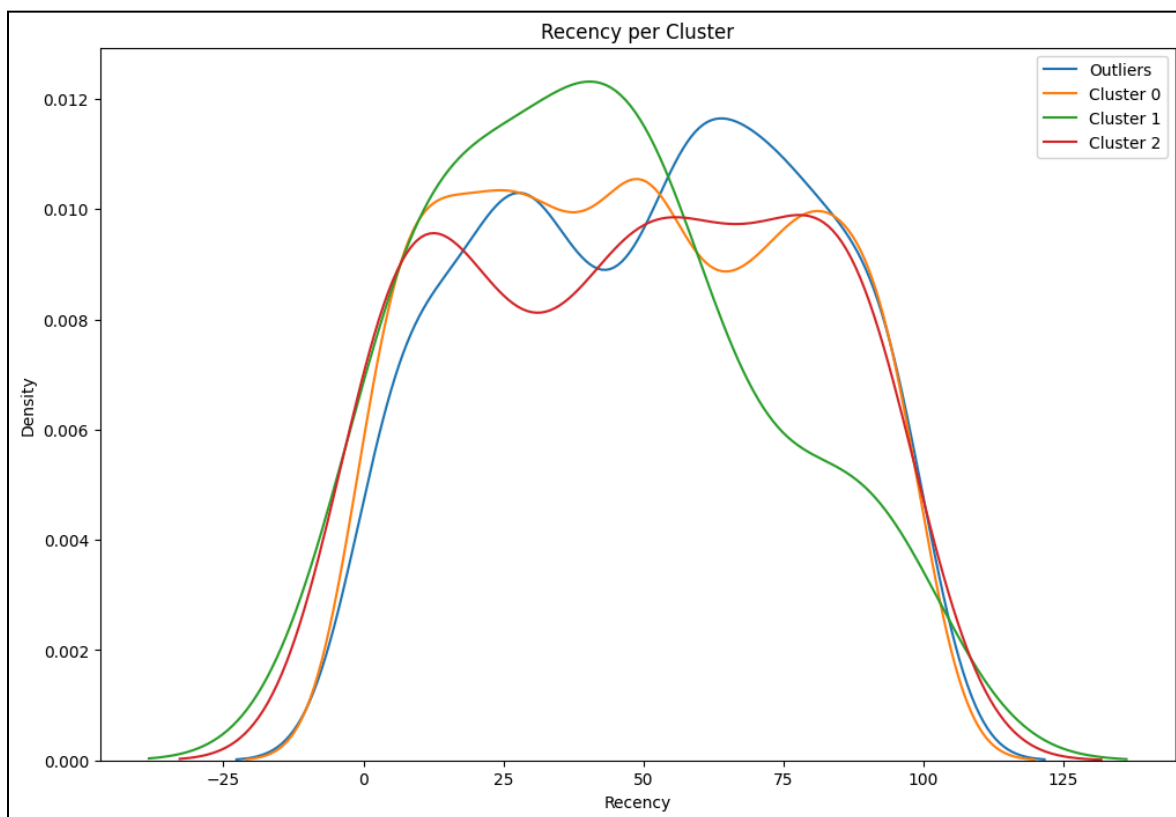
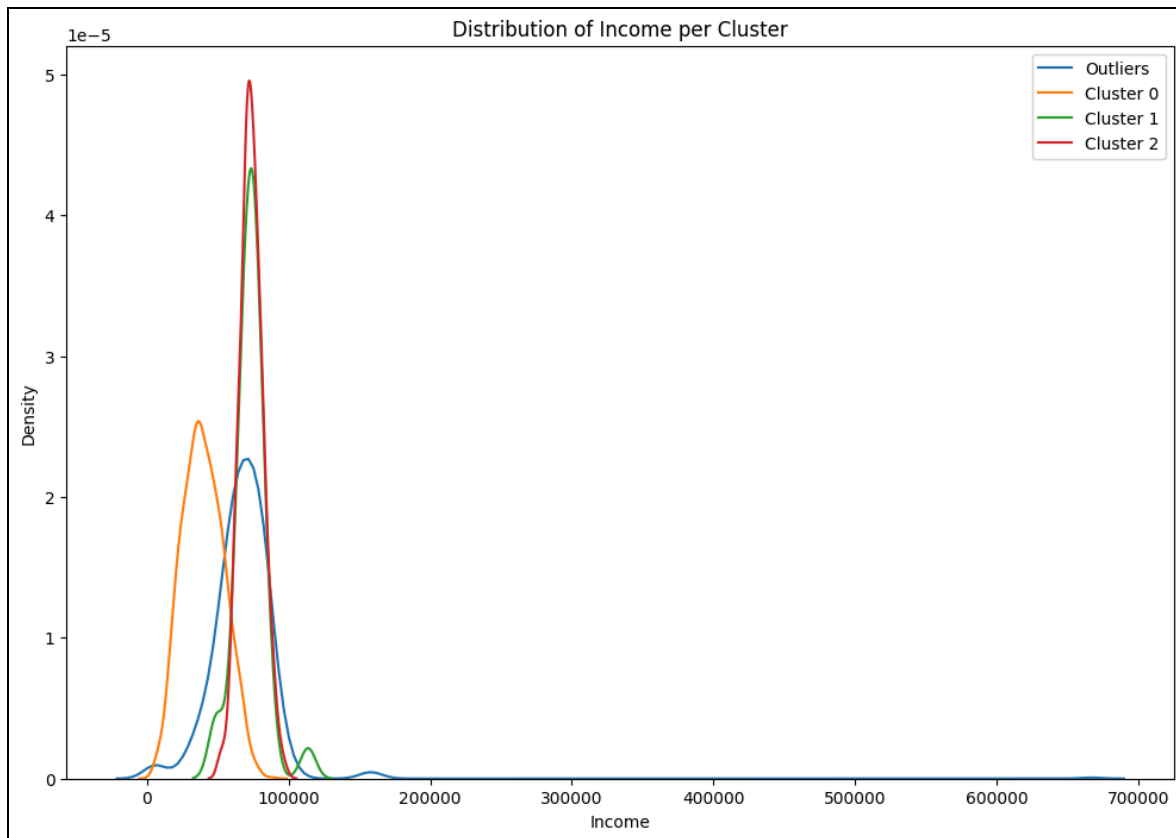
Berdasarkan tabel tersebut kita mengetahui bahwa outlier dari customer (-1) merupakan kelompok customer yang merespon tentang kampanye yang dilakukan oleh perusahaan (177). Sedangkan untuk cluster dengan customer terbanyak (0) hanya memiliki nilai respons terhadap kampanye sebesar 9%. Sehingga kita mengetahui bahwa customer yang berada pada cluster (0) tidak memberikan respon dari kampanye perusahaan.

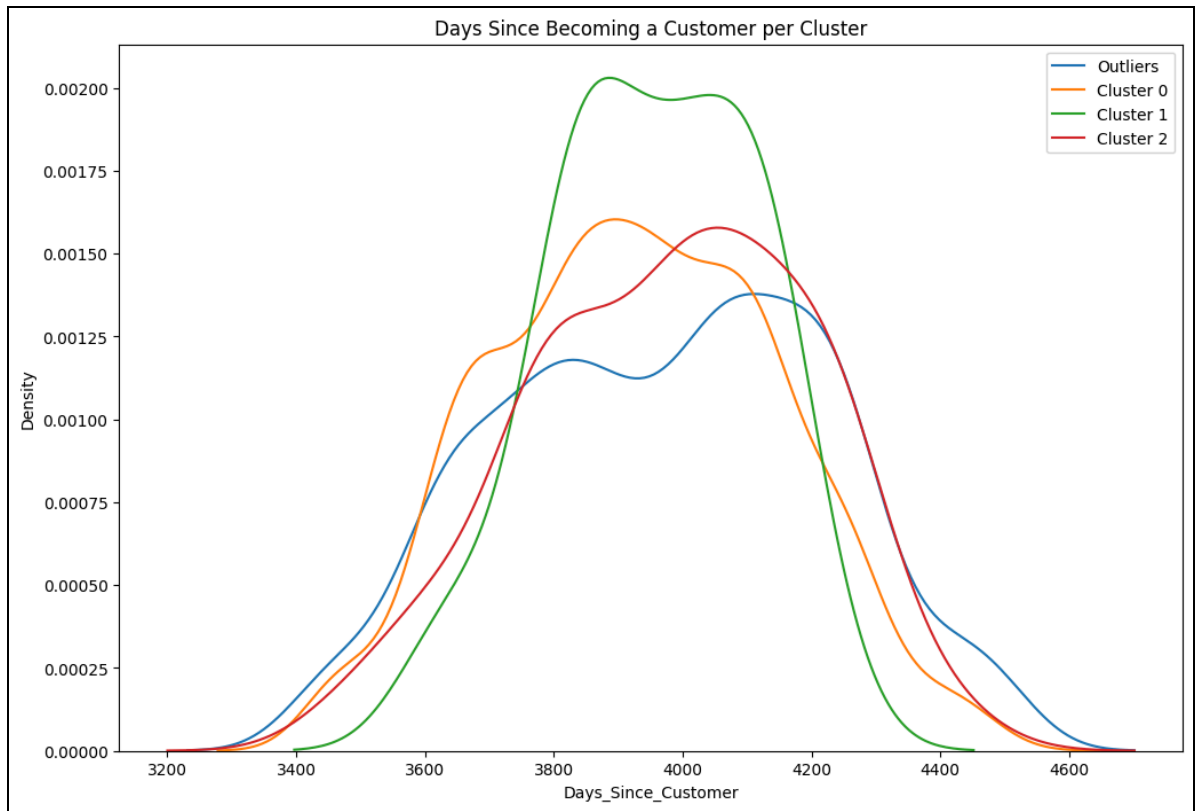
Selanjutnya kita akan mencoba menghubungkan kelompok-kelompok dari ketiga cluster customer serta outliers customer yang didapatkan dengan fitur-fitur dari sampel data.

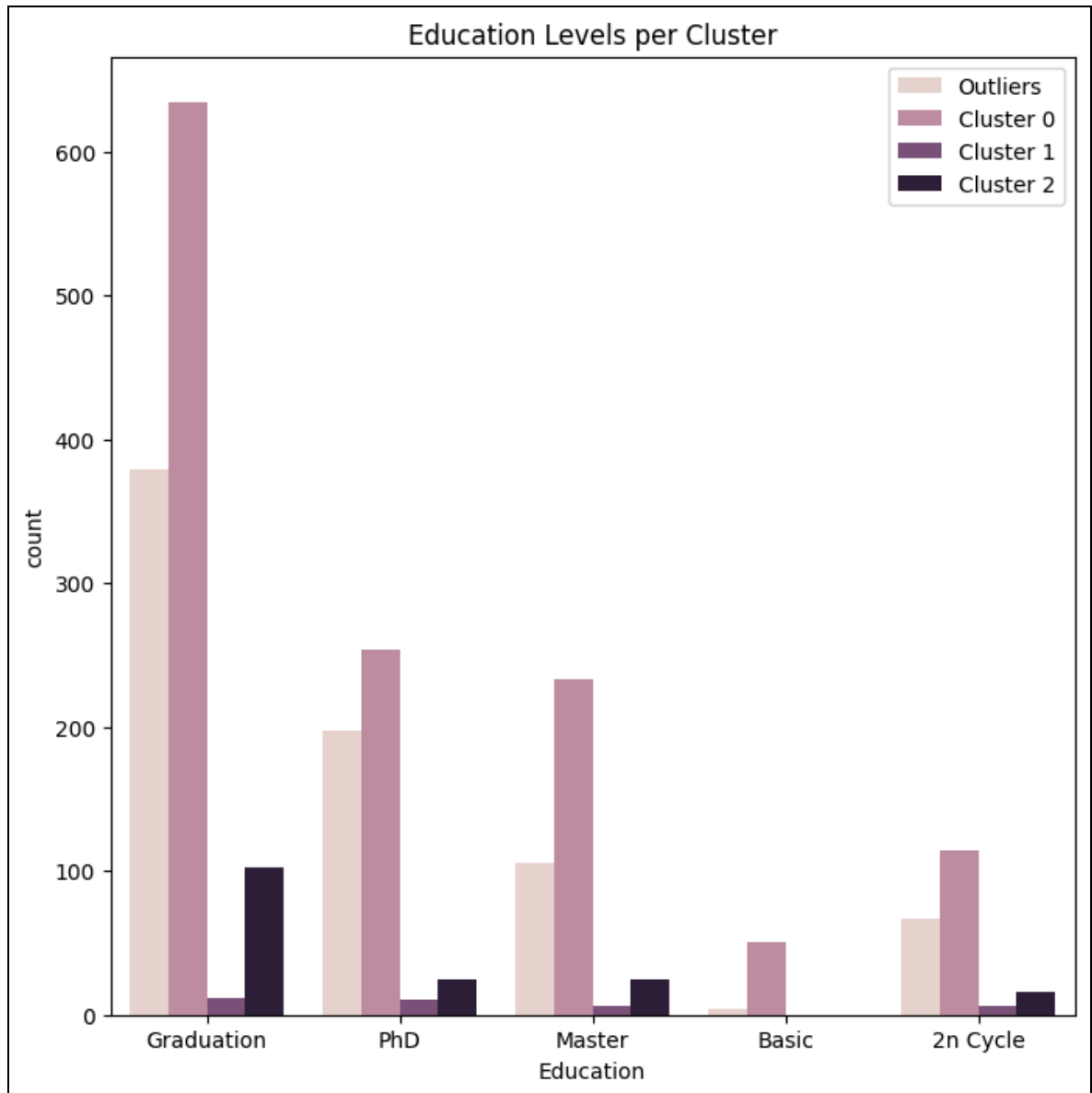


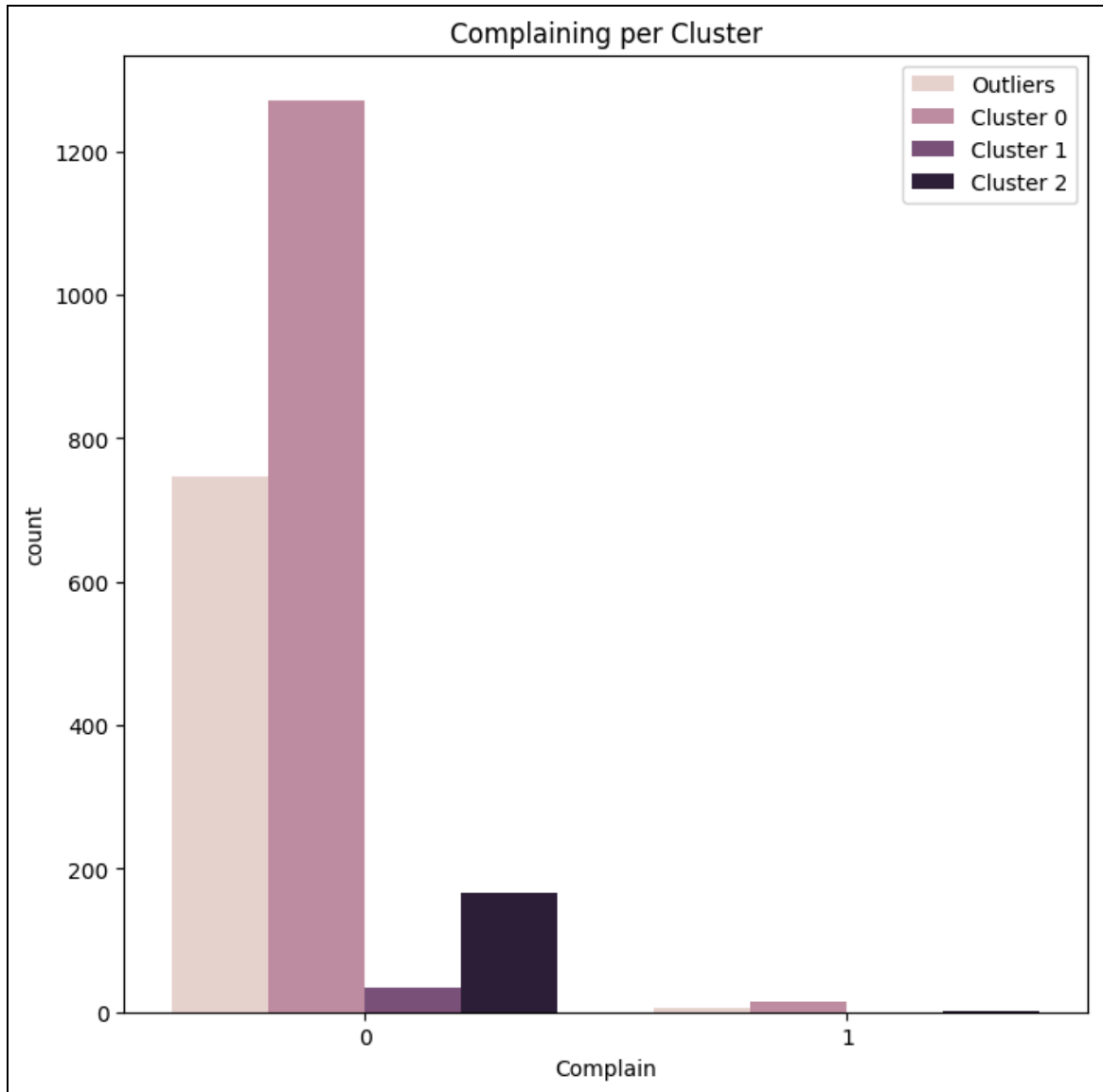












E. INTERPRETASI

Berikut adalah hasil kesimpulan yang kami temukan dengan mengimplementasikan algoritma DBSCAN pada sampel data untuk mengetahui bagaimana respon customer terhadap kampanye (campaign) yang dibuat oleh perusahaan serta bagaimana informasi dari customer untuk tiap cluster yang dihasilkan:

- **Cluster 0**
 - Rata-rata customer pada cluster ini memiliki pengeluaran yang rendah
 - Rata-rata customer pada cluster ini memiliki pendapatan yang rendah
 - Rata-rata customer pada cluster ini memiliki anggota keluarga yang banyak
 - Rata-rata customer pada cluster ini memiliki rentang usia menengah (middle age)

- Rata-rata customer pada cluster ini tidak memiliki respons terhadap kampanye pemasaran

- **Cluster 1**

- Rata-rata customer pada cluster ini memiliki pengeluaran menengah (sedang)
- Rata-rata customer pada cluster ini memiliki anggota keluarga yang sedikit
- Rata-rata customer pada cluster ini memiliki rentang usia tua (retirement age)
- Rata-rata customer pada cluster ini memiliki respons terhadap kampanye pemasaran

- **Cluster 2**

- Rata-rata customer pada cluster ini memiliki pengeluaran yang tinggi
- Rata-rata customer pada cluster ini memiliki anggota keluarga yang sedikit
- Rata-rata customer pada cluster ini memiliki rentang usia menengah (middle age)
- Rata-rata customer pada cluster ini memiliki respon yang sedikit terhadap kampanye pemasaran

- **Outliers**

- Rata-rata customer pada outlier memiliki pola pengeluaran yang bervariasi
- Rata-rata customer pada outlier memiliki pendapatan sedang - tinggi
- Rata-rata customer pada outlier memiliki anggota keluarga yang sedang
- Rata-rata customer pada outlier memiliki rentang usia pertengahan
- Rata-rata customer pada outlier memiliki respon yang tinggi terhadap pemasaran