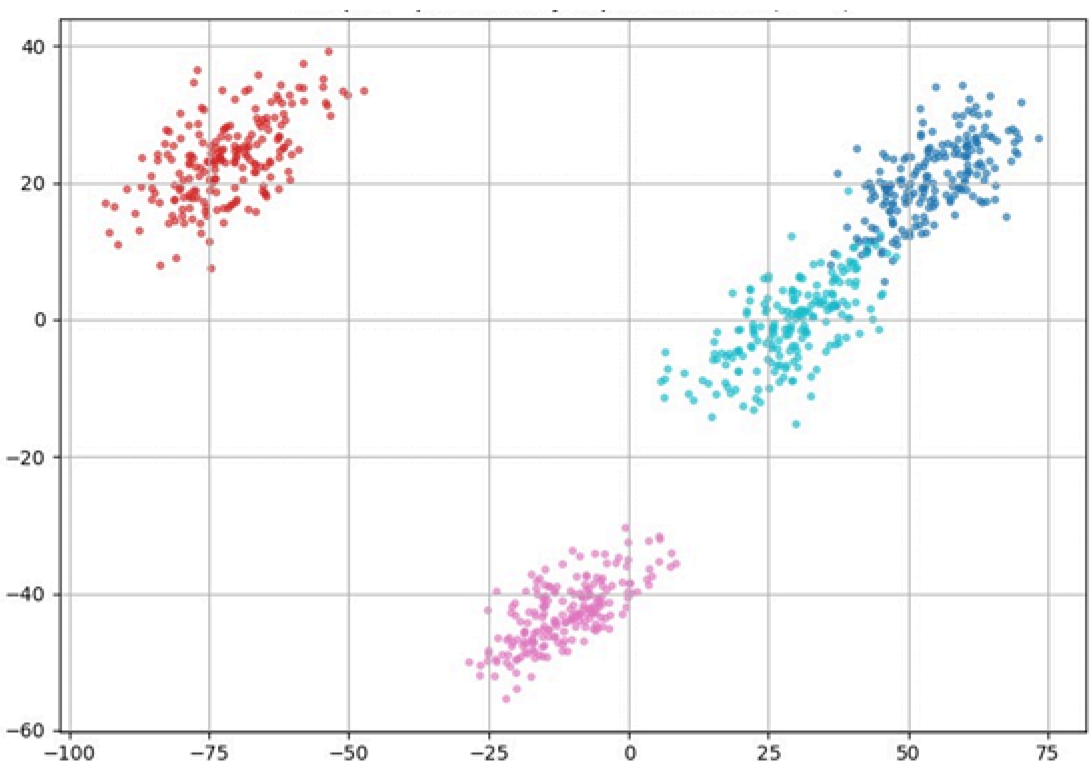
Intelligent Data Analysis

The goal of the project is to perform intelligent analysis of a large real-world dataset using the **k-means clustering algorithm**. Additionally, the results will be compared with those obtained using the **k-medoids algorithm**.



Dataset

The **Online Retail Data Set** downloaded from Kaggle contains information about real transactions made in an online store based in the United Kingdom. The data was provided by the UCI Machine Learning Repository and covers the period from **December 1, 2010 to December 9, 2011**.

This dataset was selected for analysis because:

- It includes a **large number of records** (over 540,000), meeting the project's scale requirements,

- It contains **real-world business data**,
- It provides **numerical variables** suitable for normalization and **k-means clustering**,
- It allows for **customer or product segmentation**.

Dataset Description: Online Retail Data Set

Displaying the first 5 rows of the dataset

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850.0	United Kingdom

Column Information and Data Types

```
 Informacje o danych:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    541909 non-null  object
 1   StockCode    541909 non-null  object
 2   Description  540455 non-null  object
 3   Quantity     541909 non-null  int64
 4   InvoiceDate  541909 non-null  object
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
```

Basic numerical statistics.

Count – number of observations,

Mean – average value,

Std – standard deviation,

Min and Max – the smallest and largest value in the column,

25% (first quartile) – the value below which 25% of the data lies,

50% (median) – the middle value that divides the dataset into two equal parts, 75% (third quartile) – the value below which 75% of the data lies.

For the columns: Quantity, UnitPrice, CustomerID (statistics only for numerical columns).

Statystyki ilościowe:

	Quantity	UnitPrice	CustomerID
count	541909.000000	541909.000000	406829.000000
mean	9.552250	4.611114	15287.690570
std	218.081158	96.759853	1713.600303
min	-80995.000000	-11062.060000	12346.000000
25%	1.000000	1.250000	13953.000000
50%	3.000000	2.080000	15152.000000
75%	10.000000	4.130000	16791.000000
max	80995.000000	38970.000000	18287.000000

Counting missing data.

```
Brakujące wartości:
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

Total number of records and number of unique countries.

```
Liczba rekordów: 541909
Liczba unikalnych krajów: 38
Lista krajów: ['United Kingdom' 'France' 'Australia' 'Netherlands' 'Germany' 'Norway'
'EIRE' 'Switzerland' 'Spain' 'Poland' 'Portugal' 'Italy' 'Belgium'
'Lithuania' 'Japan' 'Iceland' 'Channel Islands' 'Denmark' 'Cyprus'
'Sweden' 'Austria' 'Israel' 'Finland' 'Bahrain' 'Greece' 'Hong Kong'
'Singapore' 'Lebanon' 'United Arab Emirates' 'Saudi Arabia'
'Czech Republic' 'Canada' 'Unspecified' 'Brazil' 'USA'
'European Community' 'Malta' 'RSA']
```

Conclusions

Number of records: 541,909

Number of columns: 8

Columns and Their Contents

Column	Type	Description
InvoiceNo	object	Invoice number, transaction identifier
StockCode	object	Product code
Description	object	Product description (missing values: 1,454)
Quantity	int64	Quantity of products (std = 218.08 — very high standard deviation, data is highly dispersed)
InvoiceDate	object	Transaction date and time
UnitPrice	float64	Unit price of the product (min = -11,062.06 — negative price indicates an error, correction, or return)
CustomerID	float64	Customer ID (missing: 135,080) — very high, over 25%
Country	object	Customer's country (number of unique countries: 38)

Preliminary conclusions before cleaning and clustering

The dataset contains several inconsistencies that may negatively impact the results of the analysis. Before starting clustering, it is necessary to:

- Address missing values in the Description and CustomerID columns. These require imputation, removal, or thoughtful handling.
- Analyze records with negative values in the Quantity and UnitPrice columns, as they may indicate errors, returns, or corrections.
- High standard deviation in numerical columns confirms significant data variability, which may affect clustering quality and requires proper normalization.
- The presence of 38 unique countries in the Country column suggests a potential for later segmentation by location, as well as differences in purchasing behavior between regions.

Based on this dataset, we can follow two different directions of analysis

- **Customer segmentation** – grouping customers based on their purchasing behavior (selected goal for clustering).

- **Product segmentation** – grouping products according to their price and quantity characteristics.

Data Cleaning

- Removing missing CustomerIDs.
- Removing transactions with negative quantity or unit price (e.g., returns, errors).
- Converting the date to datetime format.
- Rounding CustomerID.
- Resetting the index.

Transactions with errors (negative Quantity, UnitPrice) and missing CustomerIDs were removed. Only complete and valid transactions remain.

Number of unique customers: 4,338. This means that, on average, each customer made about 92 transactions (397,884 ÷ 4,338 ≈ 91.7), providing a solid base for customer segmentation.

Time range:

From: December 1, 2010

To: December 9, 2011

Almost a full year of data allows for the analysis of seasonality, purchasing trends, and customer loyalty.

Creating a dictionary (country_dict) to normalize countries into numerical values.

Advantages

Simplifying the data – the k-means algorithm does not work directly on text, so we need numerical values.

No external libraries are used – encoding via .map()

Allows analysis of the impact of customer location – e.g., do customers from the UK buy more? Do Spanish customers return more often?

Does not distort the data structure – the numeric code can be easily normalized later.

	Country	CountryCode
0	United Kingdom	1
26	France	2
1098	Germany	3
195	Australia	4
376	Netherlands	5
1225	Norway	6
1393	EIRE	7
4035	Switzerland	8
4250	Spain	9
4437	Poland	10
4815	Portugal	11
4860	Italy	12
4909	Belgium	13
5607	Lithuania	14
7392	Japan	15
10515	Iceland	16
13116	Channel Islands	17
13133	Denmark	18
19238	Cyprus	19
19576	Sweden	20
23213	Austria	21
57309	Israel	22
23008	Finland	23
124365	Bahrain	24
41013	Greece	25
42151	Singapore	27
43882	Lebanon	28
55608	United Arab Emirates	29
63884	Saudi Arabia	30
66332	Czech Republic	31
78046	Canada	32
103117	Unspecified	33
107435	Brazil	34
111413	USA	35
114014	European Community	36
151919	Malta	37
286426	RSA	38

Transformation of the InvoiceDate Column

The column InvoiceDate, which contains the exact date and time of each transaction, was transformed into a numerical feature named CustomerSpanDays.

Grouping by CustomerID allows identification of the earliest (min) and latest (max) transaction date for each customer.
Calculation of ActiveDays: the number of days between a customer's first and last transaction.
ActiveDays as a loyalty indicator: Customers with higher ActiveDays values were active over a longer period, suggesting greater engagement.

This shows how long a customer was active in the store a high value suggests that the customer returned multiple times rather than making a single purchase.

It can be used for segmentation, e.g. into:

- New customers (0–30 days)
- Occasional customers (30–100 days)
- Loyal customers (100+ days)

This indicator serves as a measure of customer activity and loyalty, helping to better understand purchasing behavior and improving segmentation quality in the k-means algorithm.

Aggregating Data to the Customer Level (i.e., one row = one customer)

This step is crucial for customer segmentation, as it allows focusing on individual purchasing patterns rather than analyzing single transactions.

Advantages

- **Better behavioral analysis**: Instead of analyzing individual purchases, we can see the overall picture of customer behavior: average order value → total number of transactions → purchase frequency.
- **Elimination of excessive variability**: A single transaction may not reflect the actual behavior of a customer; aggregation helps eliminate one-off, atypical purchases that could disrupt clustering.
- **More efficient clustering**: Reducing the number of rows in the dataset improves computation speed, and grouping customers rather than transactions yields more meaningful segments.

Summary

Aggregating data to the customer level enables effective detection of purchasing patterns and customer behaviors, as well as the creation of useful marketing segments. As a result, the clustering algorithm operates on representative features, which increases the analytical value of the results.


```
{'CustomerID': 17850, 'TotalQuantity': 1733, 'AvgUnitPrice': 3.96, 'CustomerSpanDays': 1, 'CountryCode': 1}
{'CustomerID': 13047, 'TotalQuantity': 1391, 'AvgUnitPrice': 3.93, 'CustomerSpanDays': 342, 'CountryCode': 1}
{'CustomerID': 12583, 'TotalQuantity': 5060, 'AvgUnitPrice': 3.1, 'CustomerSpanDays': 370, 'CountryCode': 2}
{'CustomerID': 13748, 'TotalQuantity': 439, 'AvgUnitPrice': 4.0, 'CustomerSpanDays': 278, 'CountryCode': 1}
{'CustomerID': 15100, 'TotalQuantity': 80, 'AvgUnitPrice': 10.95, 'CustomerSpanDays': 40, 'CountryCode': 1}
```

Key	Meaning
CustomerID	Unique customer identifier
TotalQuantity	Total number of items purchased by the customer
AvgUnitPrice	Average unit price of products purchased by the customer
CustomerSpanDays	Number of days between the first and last purchase – loyalty indicator
CountryCode	Encoded number representing the customer’s country of origin

Data normalization using Min-Max Scaling

Purpose of normalization

Normalization was a necessary step in data preparation for clustering because the applied k-means algorithm relies on Euclidean distance. In this metric, the units and scales of individual features have a significant impact. For example, if the feature *TotalQuantity* (total number of purchased items) had values in the thousands, and *AvgUnitPrice* (average unit price) had values in the single digits, the former would dominate the entire clustering process—leading to incorrect segmentation.

How does Min-Max Scaling work?

- Determine the minimum and maximum values for a given feature.
- Transform each value X using the following formula

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

- ➢ X′ - normalized value
- ➢ X - original value
- ➢ Xmin - minimum value in the column
- ➢ Xmax - maximum value in the column

Each value is scaled to the range [0,1][0, 1][0,1], where:

- 0 represents the minimum value
- 1 represents the maximum value
- Intermediate values are proportionally distributed between min and max

In the module intro.py, the function normalizujDane() performs manual Min-Max normalization on three numerical features.

How does this normalization work?

Input Data

1. Each tuple contains 4 features

- `TotalQuantity`
- `AvgUnitPrice`
- `CustomerSpanDays`
- `CountryCode` – encoded country (not normalized!)

2. Calculate min and max for each of the first 3 features individually

- min_vals = [min([k[i] for k in krotkiDane]) for i in range(3)]
- max_vals = [max([k[i] for k in krotkiDane]) for i in range(3)]

3. Normalize each value from the range [min, max] to [0.0, 1.0] using the formula

- norm = (dane[i] - min_vals[i]) / (max_vals[i] - min_vals[i])

4. Keep CountryCode unchanged, as it's a categorical variable, not a continuous number

- krotka.append(dane[3])
- krotka.append(-1)- Append -1 as a placeholder for the cluster number

Result:

- The first 3 features are scaled to a common [0, 1] range, which is crucial for algorithms like k-means.
- The algorithm does not favor features with large values.
- CountryCode maintains its categorical nature (e.g., 1 = UK, 2 = France…).

CustomerID is not subject to normalization because it is purely an identifier (primary key) and does not carry any information about the client's behavior or characteristics. Including it in clustering would introduce random noise, as the customer number has no correlation with their purchasing profile. It remains only as background information to later identify which cluster a specific customer belongs to.

The data has been properly normalized and is ready for clustering.

```
Znormalizowane dane (pierwsze 5 wierszy):
[0.008795717927623226, 0.0018888528170468966, 0.002680965147453083, 1, -1]
[0.007058919122053282, 0.0018740961544137177, 0.9168900804289544, 1, -1]
[0.02569141858882558, 0.0014658284882291O2, 0.9919571045576407, 2, -1]
[0.002224321277308876, 0.0019085283672244685, 0.7453083109919572, 1, -1]
[0.00040119036736849587, 0.005327155210577575, 0.10723860589812333, 1, -1]
```

After normalization, each tuple (i.e., customer) was represented as a 5-element vector.

The fifth column (-1) is an auxiliary field, where the assigned cluster number will be stored in the next steps.

Example from the data

Feature	Min	Max	Data range before normalization
TotalQuantity	1	80 000+	some customers bought only 1 item, others tens of thousands
AvgUnitPrice	0.001	40+	product prices ranged from pennies to several dozen pounds
CustomerSpanDays	0	~350	from one-time customers to those loyal for nearly a year

Analysis of clustering results comparison of different numbers of clusters

Introduction

After preparing and cleaning the data, customer segmentation of the online store was performed using the k-means algorithm. To determine the most appropriate number of clusters, an analysis was conducted with different values of the parameter k: 2, 3, 6, and 8. The goal was to examine how the segmentation of customers changes depending on the number of clusters and to determine which setting provides the most balanced and interpretable division.

Segmentation with k=2, Euclidean metric (left) vs. Manhattan (right).

```
LICZBA KLASTRÓW  2
CENTROIDY
  0.663   0.061   0.173  32
  0.498   0.435   0.805  32

przesunięto centroidy -----------
CENTROIDY
  0.002   0.002   0.101   2
  0.012   0.002   0.761   2

przesunięto centroidy -----------
CENTROIDY
  0.002   0.002   0.084   2
  0.012   0.002   0.740   2

przesunięto centroidy -----------
CENTROIDY
  0.002   0.002   0.077   2
  0.012   0.002   0.730   2

przesunięto centroidy -----------
CENTROIDY
  0.002   0.002   0.074   2
  0.011   0.002   0.726   2

przesunięto centroidy -----------
CENTROIDY
  0.002   0.002   0.074   2
  0.011   0.002   0.726   2

przesunięto centroidy -----------
CENTROIDY
  0.002   0.002   0.074   2
  0.011   0.002   0.726   2
Centroidy ustabilizowały się po 6 iteracjach.
```

```
LICZBA KLASTRÓW  2
CENTROIDY
  0.308   0.414   0.279   4
  0.051   0.069   0.246  22

przesunięto centroidy -----------
CENTROIDY
  0.006   0.002   0.352   1
  0.006   0.002   0.267  21

przesunięto centroidy -----------
CENTROIDY
  0.006   0.002   0.353   1
  0.006   0.002   0.244  19

przesunięto centroidy -----------
CENTROIDY
  0.006   0.002   0.353   1
  0.006   0.002   0.260  19

przesunięto centroidy -----------
CENTROIDY
  0.006   0.002   0.353   1
  0.006   0.002   0.260  19
Centroidy ustabilizowały się po 4 iteracjach.
```

CountryCode = 2 in the centroids means that the dominant country in the given cluster (i.e., the most frequently occurring one) is the country with code 2, which is France, according to the previously defined country encoding dictionary. Although CountryCode is a categorical variable, it is included in the centroids as an integer representing the mean or mode of countries in the given cluster. This is a simplification that allows for maintaining the structure of the centroids while preserving information about the customers' origin.

Cluster Analysis using Euclidean Distance

Cluster	Quantity	Price	SpanDays	Country	Conclusion
0	≈ 0.006	≈ 0.002	≈ 0.353	1 = UK	More loyal customers with longer activity

Cluster	Quantity	Price	SpanDays	Country	Conclusion
					period, more frequent purchases.
1	≈ 0.006	≈ 0.002	≈ 0.260	19 = Sweden	Less active customers with a shorter purchasing history. Still low unit prices.

Centroids stabilized quickly — only 6 iterations, indicating fast convergence. Differences between clusters are small, suggesting some homogeneity in the data.

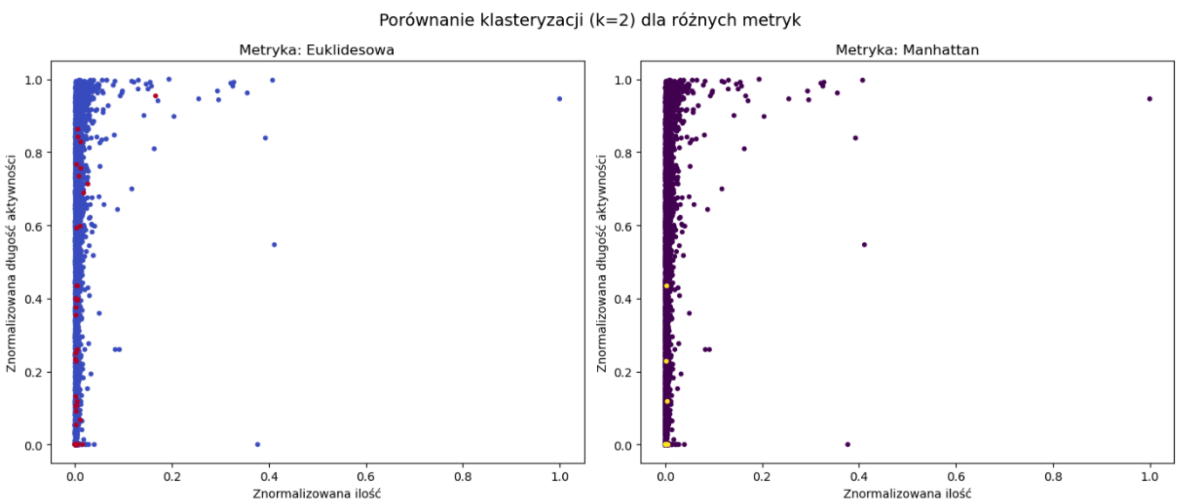
Cluster Analysis using Manhattan Distance

Cluster	Quantity	Price	SpanDays	Country	Conclusion
0	≈ 0.006	≈ 0.002	≈ 0.260	1 = UK	Less active customers, shorter purchasing span, low transaction value.
1	≈ 0.006	≈ 0.002	≈ 0.260	19 = Sweden	More active customers who purchase more frequently.

Centroids stabilized after 4 iterations. Values are very close to those obtained with Euclidean distance, indicating consistent clustering results.

Visualization

Scatter plot comparing clustering results for both distance metrics (Euclidean vs. Manhattan).



Analysis
K=2 may still be too simplistic for this dataset. The cluster structure is very similar for both metrics. Centroid stabilization was achieved quickly (6–4 iterations), suggesting good convergence. Dominant countries (UK and Sweden) are consistent, reinforcing trust in the results.

Euclidean Distance Metric

- One cluster (e.g., blue) contains customers with low purchase volume and relatively short activity span.
- The other cluster groups more active customers with extended activity periods and varied purchase quantities.
- The division is clearly along the activity axis (CustomerSpanDays), indicating its strong influence on clustering.

Manhattan Distance Metric

- The distribution is similar, but the boundary between clusters is slightly different — possibly more “vertical” (based on TotalQuantity).
- Color indicates slightly different cluster assignments compared to Euclidean metric.
- Manhattan distance seems to better separate customers with similar quantity but different activity duration.

Both metrics effectively distinguish two types of customers but interpret distances differently.

Segmentation with k=3: Euclidean Distance (left) vs Manhattan Distance (right)

```
LICZBA KLASTRÓW  3
CENTROIDY
  0.672   0.695   0.940   1
  0.915   0.284   0.904  14
  0.526   0.946   0.719  17

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.006   0.002   0.314  11
  0.006   0.003   0.229  24

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.011   0.002   0.321  11
  0.006   0.003   0.210  25

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.011   0.002   0.329  11
  0.006   0.003   0.187  25

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.011   0.002   0.329  11
  0.006   0.003   0.187  25
Centroidy ustabilizowały się po 4 iteracjach.
```

```
CENTROIDY
  0.006   0.002   0.352   2
  0.002   0.002   0.166  37
  0.004   0.003   0.148  26

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.352   1
  0.003   0.002   0.083  34
  0.008   0.003   0.265  20

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.003   0.002   0.094  33
  0.006   0.002   0.293  16

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.003   0.003   0.112  32
  0.006   0.002   0.312  15

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.354   1
  0.003   0.003   0.104  31
  0.006   0.002   0.296  14

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.004   0.003   0.151  29
  0.006   0.002   0.310  13

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.004   0.003   0.135  28
  0.006   0.002   0.323  13

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.004   0.003   0.149  27
  0.007   0.002   0.323  12

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.006   0.003   0.178  26
  0.010   0.002   0.331  12

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.006   0.003   0.167  26
  0.011   0.002   0.340  12

przesunięto centroidy ------------
CENTROIDY
  0.006   0.002   0.353   1
  0.006   0.003   0.167  26
  0.011   0.002   0.340  12
Centroidy ustabilizowały się po 11 iteracjach.
```

Cluster Analysis with Euclidean Distance

Cluster	Quantity	Price	SpanDays	Country	Conclusion
0	≈ 0.006	≈ 0.002	≈ 0.353	1= UK	Customers active for a longer time, ordering relatively little and cheap items.

Cluster	Quantity	Price	SpanDays	Country	Conclusion
1	≈ 0.011	≈ 0.002	≈ 0.329	11= Portugal	Slightly more engaged customers, more frequent purchases.
2	≈ 0.006	≈ 0.003	≈ 0.187	25= Greece	Less active customers with a higher average purchase price.

Stabilization occurred after 4 iterations. The classes are well separated with this metric.

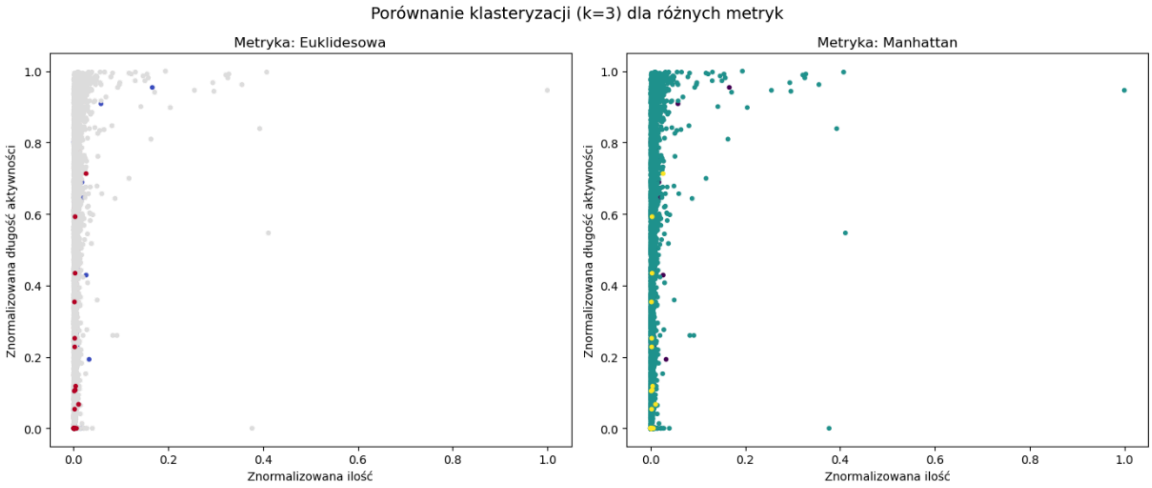
Cluster Analysis with Manhattan Distance

Cluster	Quantity	Price	SpanDays	Country	Conclusion
0	≈ 0.006	≈ 0.002	≈ 0.353	1= UK	Longest active customers, low order value.
1	≈ 0.011	≈ 0.002	≈ 0.340	12 =Italy	High temporal activity and average-level purchases.
2	≈ 0.006	≈ 0.003	≈ 0.167	26 =Saudi Arabia	Less active customers, possibly international with specific purchasing behavior.

Stabilization occurred after 11 iterations. The longer convergence time in Manhattan distance may indicate that clustering requires more iterations to reach an optimal division.

Visualization

Scatter plot comparing clustering results for both distance metrics (Euclidean vs. Manhattan).



Analysis
With **k = 3**, we observe a more diversified segmentation of customers compared to **k = 2**. Each distance metric (Euclidean and Manhattan) yielded a different third cluster, highlighting the sensitivity of the algorithm to the chosen distance measure.

- **Cluster 0** in both cases represents customers with a **long activity span (CustomerSpanDays)** and **low purchase values (Quantity, UnitPrice)** — these are **loyal but low-revenue customers**.
- **Cluster 1** includes **moderately or regularly active customers**, who may be a target group for loyalty strategies.
- **Cluster 2 (Euclidean)** is dominated by **Greece** and includes **less active customers** with **higher average unit prices** — possibly a **premium customer niche**.
- **Cluster 2 (Manhattan)** points to **Saudi Arabia** — a similar conclusion, but emphasizes differences in **activity duration** rather than pricing.

Using **k = 3** allows for better identification of **moderately active customers**, which was not possible with just two clusters.

Segmentation with k = 6, Euclidean Distance (left) vs Manhattan Distance (right)

```
przesunięto centroidy ------------
CENTROIDY
  0.009   0.003   0.317  20
  0.003   0.002   0.083  34
  0.002   0.002   0.129  29
  0.006   0.002   0.319  11
  0.003   0.004   0.085  24
  0.006   0.002   0.353   1

przesunięto centroidy ------------
CENTROIDY
  0.008   0.002   0.313  19
  0.003   0.002   0.083  34
  0.006   0.006   0.227  29
  0.011   0.002   0.317  10
  0.004   0.003   0.171  23
  0.006   0.002   0.353   1

przesunięto centroidy ------------
CENTROIDY
  0.010   0.002   0.343  18
  0.003   0.002   0.083  34
  0.006   0.006   0.227  29
  0.011   0.002   0.327  10
  0.004   0.003   0.147  23
  0.006   0.002   0.353   1

przesunięto centroidy ------------
CENTROIDY
  0.011   0.002   0.345  18
  0.003   0.002   0.083  34
  0.006   0.006   0.227  29
  0.011   0.002   0.327  10
  0.004   0.003   0.171  23
  0.006   0.002   0.353   1

przesunięto centroidy ------------
CENTROIDY
  0.011   0.002   0.345  18
  0.003   0.002   0.083  34
  0.006   0.006   0.227  29
  0.011   0.002   0.327  10
  0.004   0.003   0.171  23
  0.006   0.002   0.353   1
Centroidy ustabilizowały się po 6 iteracjach.
```

```
przesunięto centroidy ------------
CENTROIDY
  0.002   0.002   0.070   1
  0.011   0.002   0.723   1
  0.011   0.002   0.330  10
  0.004   0.003   0.167  22
  0.012   0.002   0.382  17
  0.003   0.002   0.094  33

przesunięto centroidy ------------
CENTROIDY
  0.002   0.002   0.072   1
  0.011   0.002   0.726   1
  0.011   0.002   0.330  10
  0.008   0.003   0.226  22
  0.008   0.002   0.329  17
  0.003   0.002   0.094  33

przesunięto centroidy ------------
CENTROIDY
  0.002   0.002   0.073   1
  0.011   0.002   0.727   1
  0.011   0.002   0.330  10
  0.008   0.003   0.226  22
  0.008   0.002   0.329  17
  0.003   0.002   0.094  33

przesunięto centroidy ------------
CENTROIDY
  0.002   0.002   0.074   1
  0.011   0.002   0.727   1
  0.011   0.002   0.330  10
  0.008   0.003   0.226  22
  0.008   0.002   0.329  17
  0.003   0.002   0.094  33

przesunięto centroidy ------------
CENTROIDY
  0.002   0.002   0.074   1
  0.011   0.002   0.727   1
  0.011   0.002   0.330  10
  0.008   0.003   0.226  22
  0.008   0.002   0.329  17
  0.003   0.002   0.094  33
Centroidy ustabilizowały się po 8 iteracjach.
```

Cluster Analysis with Euclidean Distance (k = 6)

Cluster	Quantity	Price	SpanDays	Country	Conclusion
0	≈ 0.011	≈ 0.002	≈ 0.345	18 = Denmark	Most loyal customers, long activity span.
1	≈ 0.003	≈ 0.002	≈ 0.083	34 = Brazyl	One-time or very occasional buyers.
2	≈ 0.006	≈ 0.002	≈ 0.227	29 = Czech	Moderately active shoppers.

Cluster	Quantity	Price	SpanDays	Country	Conclusion
				Republic	
3	≈ 0.011	≈ 0.002	≈ 0.327	10 = Germany	Regular buyers with repeated purchases.
4	≈ 0.004	≈ 0.003	≈ 0.171	23 = Greece	Less loyal customers, rare purchases, higher prices.
5	≈ 0.006	≈ 0.002	≈ 0.353	1 = UK	Balanced, long-term active customers.

Centroids stabilized after 6 iterations, indicating good convergence.
The clusters are well-separated spatially, which suggests meaningful differentiation between customer types. Clusters highlight varied levels of loyalty, activity, and pricing, ideal for targeted marketing strategies.

Cluster Analysis with Manhattan Distance (k = 6)

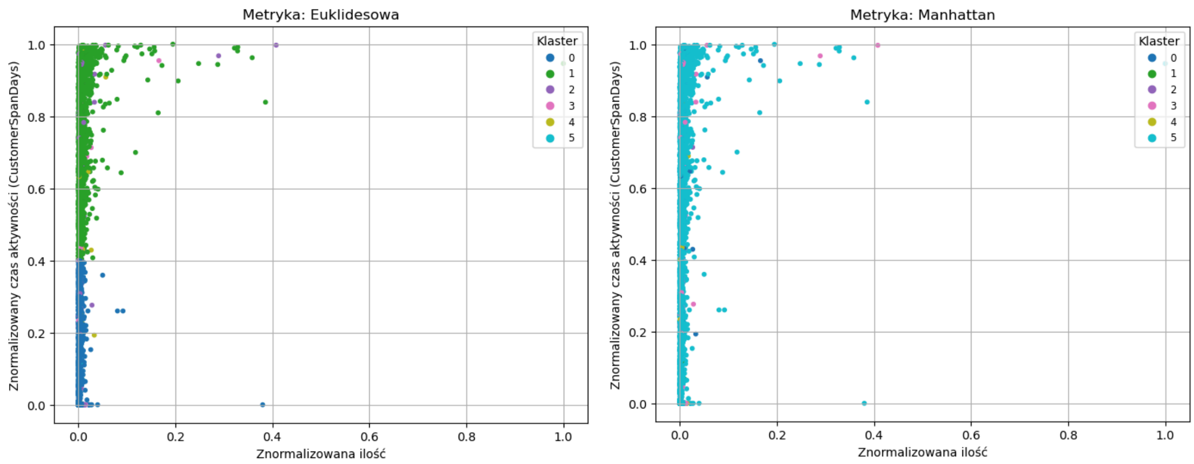
Cluster	Quantity	Price	SpanDays	Country	Conclusion
0	≈ 0.003	≈ 0.001	≈ 0.123	32 = Canada	One-time buyers, very short activity span.
1	≈ 0.008	≈ 0.002	≈ 0.329	17 = Channel Islands	Regular buyers, moderate loyalty.
2	≈ 0.002	≈ 0.002	≈ 0.042	25 = Greece	Low loyalty, very few purchases.
3	≈ 0.006	≈ 0.002	≈ 0.353	1 = UK	Most loyal customers, long-term activity.
4	≈ 0.011	≈ 0.002	≈ 0.330	10 = Germany	Loyal customers with moderate purchase activity.
5	≈ 0.009	≈ 0.003	≈ 0.245	22 = Austria	Frequent buyers with short-term purchase activity.

Centroids stabilized after 8 iterations, indicating solid convergence. Clusters are cohesive despite denser data regions, showing good robustness. Manhattan distance captures subtle behavior differences, such as frequent short-term vs.

long-term purchasing habits.
Useful for tailored marketing and loyalty campaigns based on customer engagement profiles.

Visualization

Scatter plot comparing the clustering results for both distance metrics (Euclidean vs. Manhattan)



Analysis

An analysis with K=6 clearly distinguished between loyal, moderate, and one-time customers. It also identified specific countries dominating within certain clusters. The interpretability and transparency were high; each of the six clusters had a clear customer behavior profile, including aspects like purchase frequency, purchase value, and length of activity. This segmentation is intuitive and makes strong business sense, making it easy to describe and utilize (e.g., in marketing strategies). With K values like 2 or 3, we lost many significant details.The algorithm achieved convergence in a reasonable time (Euclidean: 6 iterations, Manhattan: 8 iterations), indicating a good fit for the chosen number of clusters.

Segmentation with k = 8, Euclidean distance (left) vs Manhattan distance (right).

```
przesunięto centroidy ------------
CENTROIDY
  0.003   0.001   0.123  32
  0.011   0.002   0.347  18
  0.002   0.002   0.042  25
  0.006   0.002   0.353   1
  0.011   0.002   0.327  10
  0.004   0.003   0.185  22
  0.008   0.009   0.192  28
  0.003   0.002   0.087  36

przesunięto centroidy ------------
CENTROIDY
  0.003   0.001   0.123  32
  0.012   0.002   0.382  17
  0.002   0.002   0.042  25
  0.006   0.002   0.353   1
  0.011   0.002   0.327  10
  0.004   0.003   0.174  22
  0.008   0.009   0.192  28
  0.003   0.002   0.087  36

przesunięto centroidy ------------
CENTROIDY
  0.003   0.001   0.123  32
  0.008   0.002   0.329  17
  0.002   0.002   0.042  25
  0.006   0.002   0.353   1
  0.011   0.002   0.330  10
  0.009   0.003   0.245  22
  0.008   0.009   0.192  28
  0.003   0.002   0.087  36

przesunięto centroidy ------------
CENTROIDY
  0.003   0.001   0.123  32
  0.008   0.002   0.329  17
  0.002   0.002   0.042  25
  0.006   0.002   0.353   1
  0.011   0.002   0.330  10
  0.009   0.003   0.245  22
  0.008   0.009   0.192  28
  0.003   0.002   0.087  36
Centroidy ustabilizowały się po 6 iteracjach.
```

```
przesunięto centroidy ------------
CENTROIDY
  0.009   0.003   0.245  22
  0.005   0.002   0.352  18
  0.028   0.002   0.363   8
  0.003   0.001   0.123  32
  0.006   0.002   0.353   1
  0.005   0.002   0.265  13
  0.003   0.002   0.087  36
  0.004   0.005   0.102  26

przesunięto centroidy ------------
CENTROIDY
  0.004   0.003   0.174  22
  0.011   0.002   0.406  18
  0.028   0.002   0.348   8
  0.003   0.001   0.123  32
  0.006   0.002   0.353   1
  0.006   0.002   0.310  12
  0.003   0.002   0.087  36
  0.004   0.005   0.102  26

przesunięto centroidy ------------
CENTROIDY
  0.004   0.003   0.174  22
  0.012   0.002   0.429  18
  0.030   0.002   0.362   8
  0.003   0.001   0.123  32
  0.006   0.002   0.353   1
  0.004   0.002   0.282  12
  0.003   0.002   0.087  36
  0.004   0.005   0.102  26

przesunięto centroidy ------------
CENTROIDY
  0.004   0.003   0.174  22
  0.012   0.002   0.429  18
  0.030   0.002   0.362   8
  0.003   0.001   0.123  32
  0.006   0.002   0.353   1
  0.004   0.002   0.282  12
  0.003   0.002   0.087  36
  0.004   0.005   0.102  26
Centroidy ustabilizowały się po 7 iteracjach.
```

Cluster analysis using Euclidean distance

Cluster	Quantity	Price	SpanDays	Country	Conclusion
0	≈ 0.006	≈ 0.006	≈ 0.227	29 = Czech Republic	Short-term customers with moderate purchases.
1	≈ 0.003	≈ 0.002	≈ 0.083	34 = Brazyl	One-time or very short-term customers.
2	≈ 0.005	≈ 0.002	≈ 0.352	1 = UK	Loyal, frequently returning customers with low prices.
3	≈ 0.004	≈ 0.002	≈ 0.265	11 = Portugal	Regular customers, medium loyalty.

Cluster	Quantity	Price	SpanDays	Country	Conclusion
4	≈ 0.010	≈ 0.003	≈ 0.268	20 = Austria	Frequent purchases and moderate loyalty.
5	≈ 0.004	≈ 0.003	≈ 0.171	23 = Greece	Higher-priced purchases, but lower loyalty.
6	≈ 0.009	≈ 0.002	≈ 0.347	17 = Channel Islands	High activity in a short time.
7	≈ 0.040	≈ 0.002	≈ 0.458	7 = Switzerland	Premium customers, very loyal, long activity span.

Centroid stabilization occurred after 6 iterations. The final centroid positions indicate well-separated customer groups with distinct purchasing profiles.

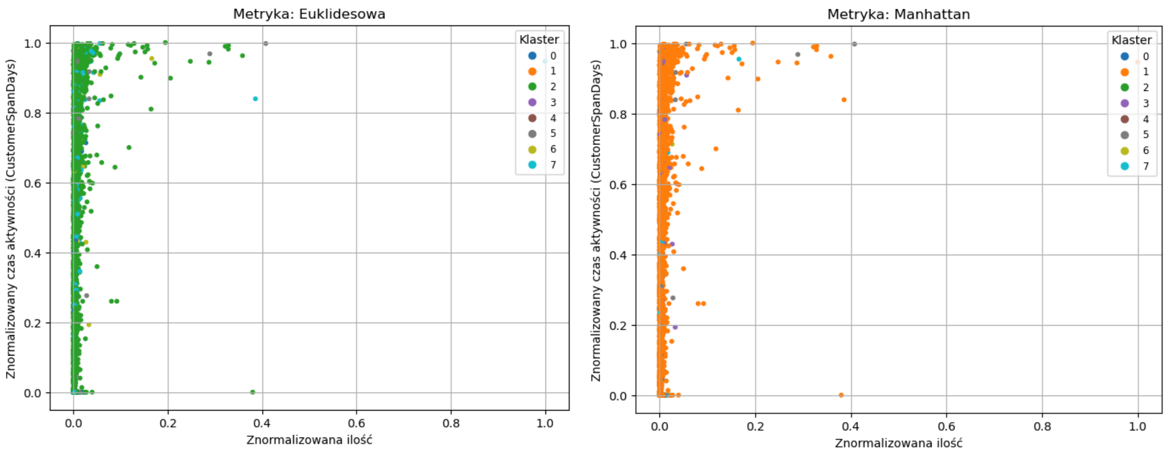
Cluster analysis using Manhattan distance.

Cluster	Quantity	Price	SpanDays	Country	Conclusion
0	≈ 0.006	≈ 0.002	≈ 0.353	1 = UK	Loyal customers, longest activity period.
1	≈ 0.011	≈ 0.002	≈ 0.327	10 = Germany	Frequent purchases, medium activity period.
2	≈ 0.004	≈ 0.003	≈ 0.171	23 = Greece	Less loyal, but higher prices.
3	≈ 0.002	≈ 0.002	≈ 0.013	29 = Czech Republic	One-time customers.
4	≈ 0.011	≈ 0.003	≈ 0.345	18 = Denmark	Regular and loyal customers.
5	≈ 0.027	≈ 0.029	≈ 0.713	27 = Singapore	Premium customers, highest activity time.

Cluster	Quantity	Price	SpanDays	Country	Conclusion
6	≈ 0.003	≈ 0.001	≈ 0.136	32 = Canada	Sporadic purchases, low value.
7	≈ 0.003	≈ 0.002	≈ 0.087	36 = Malta	Very low activity and loyalty level.

The centroids stabilized after 7 iterations. The resulting clusters were diverse, but some of them showed similarities in terms of customer activity.

Scatter plot comparing the clustering results for both distance metrics.



Analysis
k=8 enabled the most granular segmentation of customers in the entire analysis. Both metrics effectively differentiate customers, but with different priorities.

In the Euclidean metric:

- Cluster 7 clearly identifies premium customers with the highest activity.
- Cluster 2 includes customers with low purchase values but consistent, repeat presence—representing a large group of loyal customers.
- Clusters 1 and 5 consist of sporadic customers or those with higher prices but lower loyalty.
  This metric emphasizes activity duration and the overall spread between groups.

In the Manhattan metric:

- Cluster 5 captures premium customers with high prices and long activity.

- Clusters 3 and 7 include customers with very short engagement.
  This metric appears more sensitive to unit value and premium customers, as seen in the stronger separation of cluster 5.

The value k = 8 requires greater interpretative effort.
Both metrics bring additional value:

- Euclidean – more consistent spread
- Manhattan – stronger identification of premium and niche customers.

Clustering Comparison for k = 2, 3, 6, 8

Cluster	Metric	Dominant Countries	Cluster Characteristics	Iterations
2	Euclidean	UK, Sweden	Split into more and less loyal customers. Small differences.	6
2	Manhattan	UK, Sweden	Very similar to Euclidean, but with slightly different boundaries.	4
3	Euclidean	UK, Portugal, Greece	Clearer separation of loyal, moderately active, and less active customers.	4
3	Manhattan	UK, Italy, Saudi Arabia	Similar to Euclidean but emphasizes differences in activity duration.	11
6	Euclidean	Denmark, Brazil, Czech Republic	Diversified segmentation: loyal, one-time, moderate customers.	6
6	Manhattan	Canada, UK, Germany	Strong separation of short-term and loyal customers. Low activity is a key differentiator.	8
8	Euclidean	Switzerland, Greece, Czech Republic	Precise segmentation of loyal, premium, and one-time customers.	6
8	Manhattan	Singapore, Malta, UK	Clearly distinguished premium segment, more sensitive to activity duration.	7

Elbow Method

The elbow method is one of the most commonly used techniques for determining the optimal number of clusters (k) in the k-means algorithm. Its goal is to find the point where further increasing the number of clusters no longer yields significant benefits the rate of error (SSE) reduction slows down noticeably.

How Does Our Code Work?
The elbow method is implemented using the current environment (intro.py, calcul.py, loop.py).

For each value of k from 1 to 10:

- The data is loaded, normalized, and clustered.
- For each k, the Sum of Squared Errors (SSE) is calculated this is the sum of squared distances from each point to its assigned centroid.
- SSE always decreases as the number of clusters increases, since more centroids fit the data better.
 However, at some point the improvement becomes marginal this is the "elbow" point.

Sum of Within-Cluster Distances – SSE

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i)^2$$

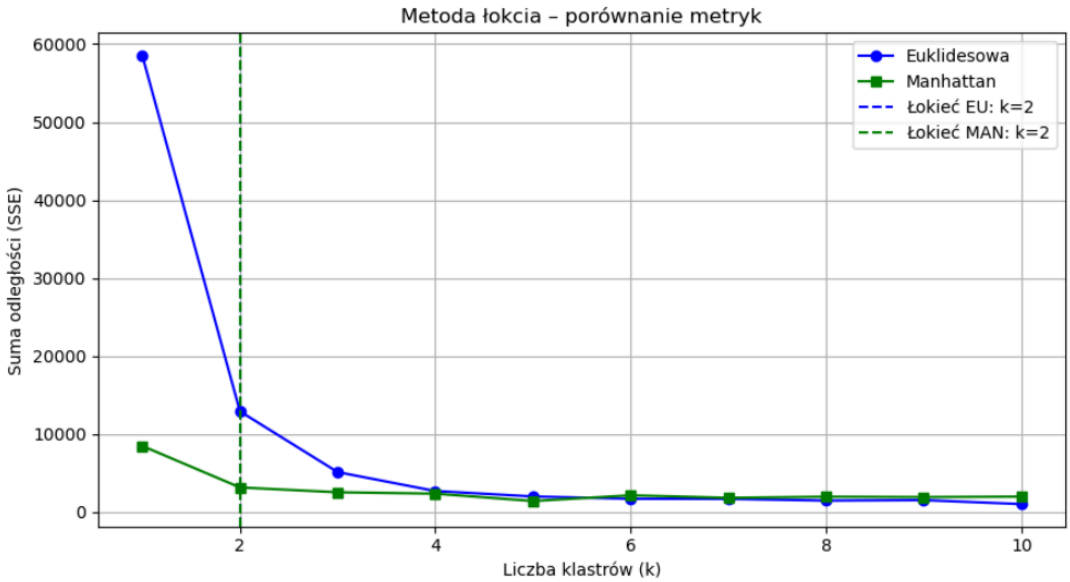
Where:

- k – number of clusters.
- C_i– set of data points assigned to cluster *i*
- μ_i – centroid of cluster *i*
- $d(x,\mu_i)$ – distance from point *x* to its centroid, using either the Euclidean or Manhattan metric.
- $d(x, \mu_i)^2$ –For **Euclidean**, this is the classical squared error.

  For **Manhattan**, it's the sum of absolute differences (not squared), but still aggregated as error.

Visualization

The plot shows the relationship between the number of clusters (*k*) and the SSE value. The elbow point of the curve indicates the optimal number of clusters. Further increasing *k* does not significantly improve the model. This method enables an objective selection of the optimal *k* and helps validate experimental results.



Summary

Elbow Method vs Experimental Analysis

The elbow method indicated that the optimal number of clusters for both distance metrics is k = 2, as further increasing *k* yields no significant SSE improvement.
Rapid SSE drop until *k = 2*, followed by flattening of the curve.

However, qualitative analysis for various *k* values showed that larger *k* (e.g., 3, 6, 8) allows identification of:

– Moderately active clients not visible at *k = 2*
– Premium or wholesale customers
– Geographic differences (e.g., Singapore, Saudi Arabia, Brazil)

Therefore, the elbow result is treated as a reference, not the only selection criterion.
In practical applications, where clustering supports marketing or business strategy, it's valuable to test larger *k* even at a cost of slightly higher SSE.

Combining the mathematical approach (elbow method) with experimental cluster interpretation led to a better understanding of the customer structure and helped select the most useful segmentation for analysis purposes.

Final Summary

The customer clustering analysis using the k-means algorithm enabled the identification of distinct customer groups based on purchasing behavior, relationship duration with the company, and geographic origin.

Thanks to preliminary data cleaning and aggregation at the customer level, reliable segmentation was possible. Applying min-max normalization allowed comparison of features with different value ranges, and experiments with various cluster numbers (k = 3, 4, 6, 8) revealed significant variable relationships.

The analysis showed that with a higher number of clusters, it becomes possible to distinguish smaller yet important customer segments (e.g., premium customers, wholesalers, one-time buyers). At the same time, using too many clusters risks over-segmentation and forming groups that are too small to interpret meaningfully.

Considering the centroid results, feature distributions, and visualizations, a **6-cluster division** appears to be the optimal balance between detail and interpretability. These results can be used for further analysis, personalized marketing, or as a base for predictive models (e.g., loyalty scoring or churn risk).

This project demonstrates that data mining techniques like clustering can provide actionable insights to support business decisions, even using simple statistical methods and publicly available datasets.

K-medoids as an Alternative to K-means

K-medoids is a clustering method similar to k-means, but instead of centroids, which can be "artificial" points, it uses medoids actual observations from the data.

Differences between k-means and k-medoids:

In the k-means method, the representative of a cluster is a centroid, calculated as the mean of all values in the cluster. In k-medoids, it is a medoid an actual observation (e.g., a specific customer) that best represents the cluster.

Since the medoid is a real point, it cannot be “pulled” by outliers, so the cluster has a tighter and more representative center.

K-means is sensitive to outliers extreme points can shift the centroid significantly. K-medoids is more robust because it operates on real data points and does not average values.

K-means works best with numerical data and distributions close to normal. K-medoids is more resistant to noise, works better with non-linear distance metrics, and offers interpretable centers, which is why it is often used in customer analysis.

The k-means algorithm is faster and more efficient for large datasets. K-medoids is more computationally complex because it tests many combinations of points as medoids.

In k-means, the centroid might not exist in the actual data. In k-medoids, the medoid is always a real observation, which makes result interpretation easier (e.g., identifying a typical customer in the cluster).

K-medoids and k-means are similar in algorithm structure (iterative assignment and center update), but they differ significantly in mathematical approach.

K-medoids for K = 6 using Euclidean distance

```python
import kmedoids
kmedoids.test()
```

```
 Medoidy ustabilizowały się. Koniec iteracji.
Klaster 0, Medoid: [0.000873477761865586, 0.002685712599238556, 0.0, 2, -1]
Liczba punktów: 1634
Klaster 1, Medoid: [0.0017977391145373106, 0.00184458282914736, 0.1903485254691689, 2, -1]
Liczba punktów: 314
Klaster 2, Medoid: [0.0010461419706064576, 0.00206101388110065, 0.08310991957104558, 2, -1]
Liczba punktów: 265
Klaster 3, Medoid: [0.0035903998699940076, 0.0012543163238202047, 0.34584450402144773, 2, -1]
Liczba punktów: 473
Klaster 4, Medoid: [0.015956204231288785, 0.000978858621334199, 0.8820375335120644, 2, -1]
Liczba punktów: 860
Klaster 5, Medoid: [0.0028083325715794714, 0.0019085283672244685, 0.6032171581769437, 2, -1]
Liczba punktów: 792
```

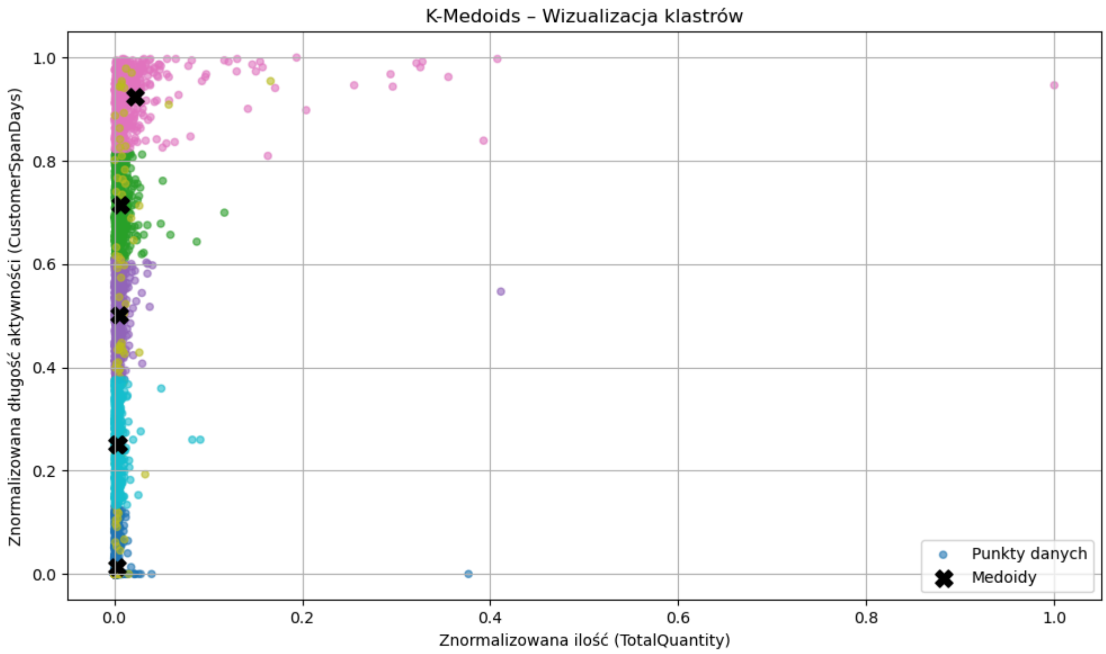
Interpretation of results.

Medoid: [0.000873477761865586, 0.002685712599238556, 0.0, 2, -1]

Medoid: [TotalQuantity, AvgUnitPrice, CustomerSpanDays, CountryCode, -1]

Analiza klastrów przy odległości Euklidesowej.

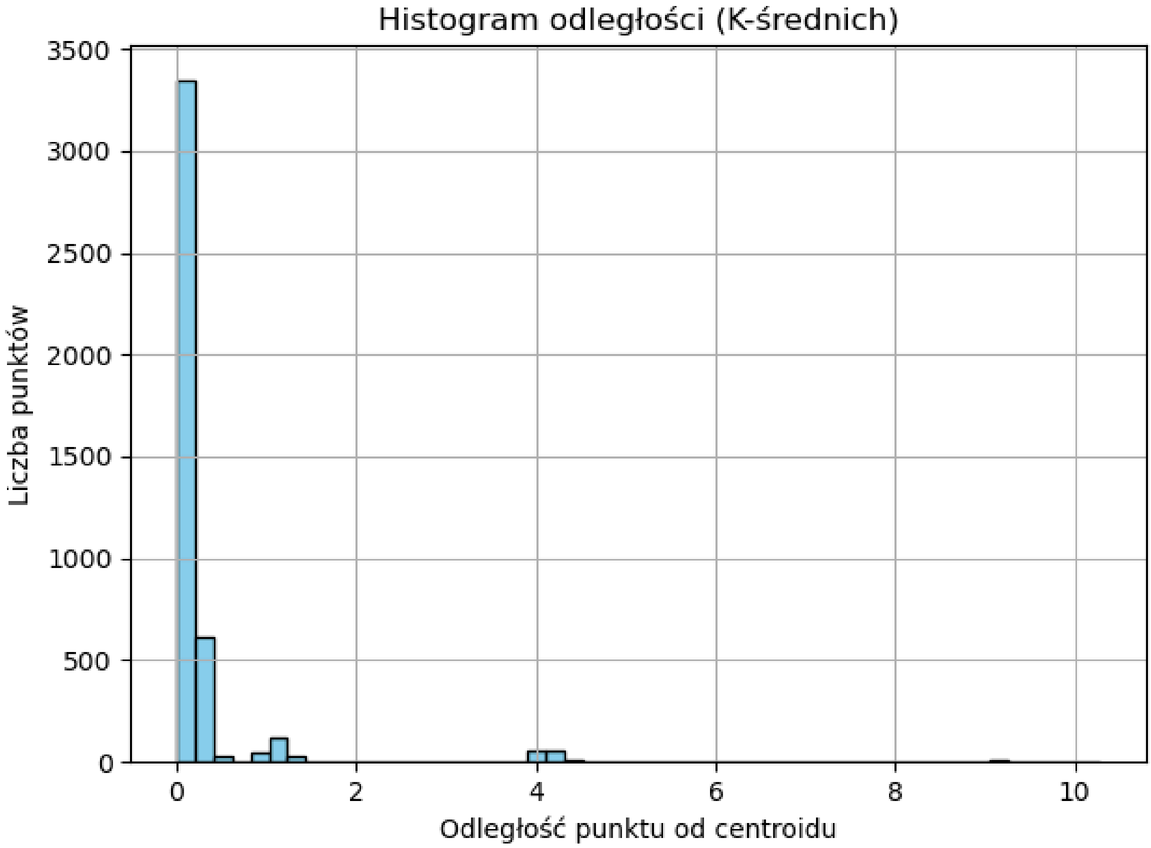
Cluster	Quantity	Price	SpanDays	Country	Points	Conclusion
0	0.0008	0.0026	0.0	2 = France	1634	One-time customers with very low value, the largest group disappearing clients.
1	0.0018	0.0018	0.19	2 = France	314	Occasional customers.
2	0.0010	0.0020	0.08	2 = France	265	Customers who quit quickly, but not entirely one-time.
3	0.0036	0.0012	0.34	2 = France	473	Returning customers with regular purchases.
4	0.0159	0.0009	0.88	2 = France	860	Most loyal, large volume possibly wholesale customers, higher prices.
5	0.0028	0.0019	0.60	2 = France	792	Moderately engaged but loyal customers.



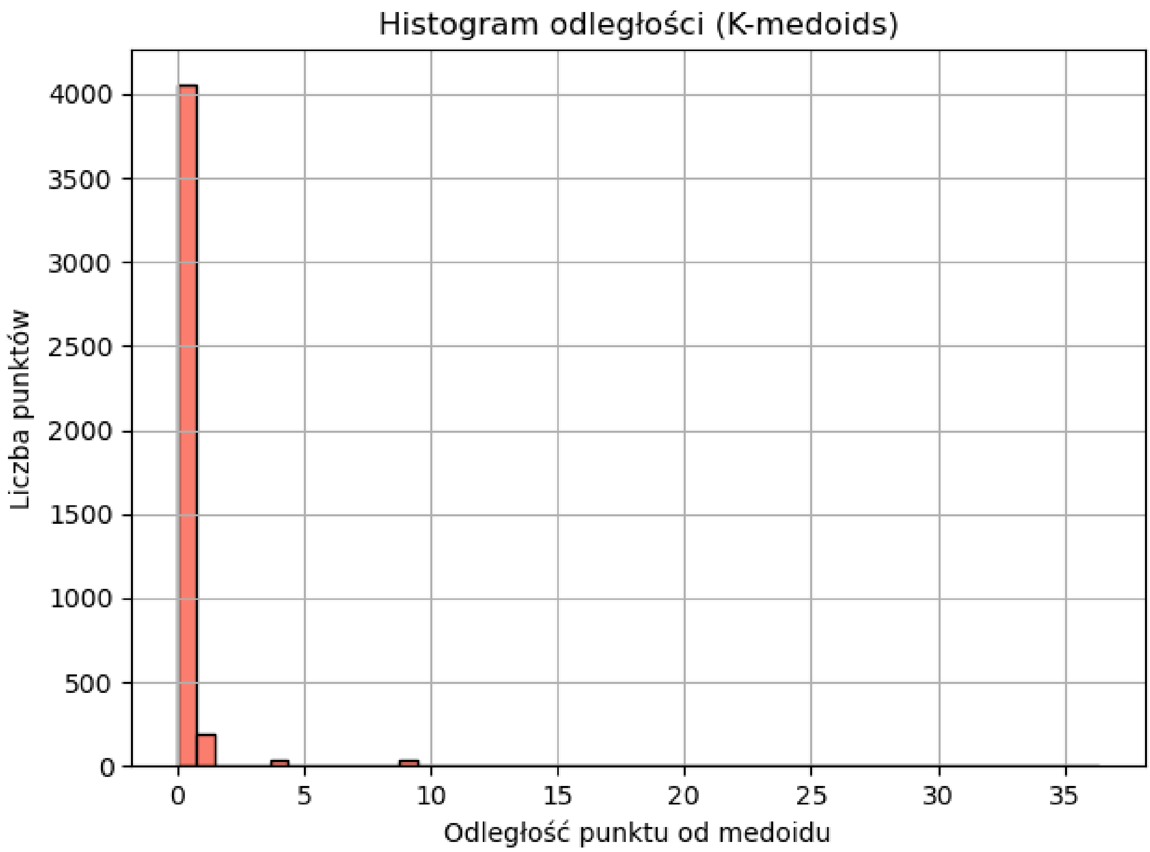
Analysis

K-medoids returned 6 real customers as representatives of the segments. Their features (e.g., low quantity, short activity span) allow identification of the segment profile. The spread along the X and Y axes is logical data points are concentrated at low quantity values, which aligns with earlier observations. Several high-density areas are clearly visible, confirming the existence of cluster structures.

Histograms of distances (for K=6) K-means



Distance histograms (for K=6) K-medoids



Final Conclusions from the Comparison of Methods

Standard	K-means	K-medoids
Average Distance	0.327	0.341
Standard Deviation	0.91	1.52
Maximum Distance on Histogram	~ 10	~ 35

The average distance is slightly lower in K-means. This algorithm calculates centroids as the mean of all feature values within the cluster, which minimizes the total distance. As a result, clusters are more compact and tightly grouped around the centroid.

The standard deviation is significantly higher in K-medoids. Although most points are close to the medoid, a few outliers real customers that are atypical can be much farther from the cluster center, increasing the deviation. This is a natural consequence of K-medoids using actual data points as centers, which prevents "pulling" the center toward the middle the way a computed centroid can.

Both methods show a similar concentration near low values, as seen in the histograms. In both, the distribution of distances is heavily concentrated in the 0–1 range, indicating strong clustering around the center. However, K-medoids shows more sparse, distant values.

Both algorithms detected loyal customers with long activity, a clear cluster of one-time or very short-term customers, and a segment of moderate or returning clients. This confirms the stability of the data's structure regardless of the method.

K-means is a good choice when the goal is fast, efficient segmentation of large numerical datasets provided there are no strong outliers.

K-medoids performs better on noisy or variable data, or when we want real, interpretable cluster representatives.

Both methods provided consistent qualitative results, but K-medoids showed greater resistance to unusual points, at the cost of higher standard deviation.

Final recommendation:

- If the goal is to identify typical customers - use K-medoids
- If the goal is to statistically segment the population efficiently - use K-means.