

Customer Segmentation with k-Means for Demo Online Shop

Name: Johann Zahlmann

Date: 31.03.2024

Table of Contents

LIST OF FIGURES	2
1 BUSINESS UNDERSTANDING.....	3
2 DATASET & DATA CLEANING	5
3 SELECTION OF FUNNEL PHASE AND FEATURE SELECTION.....	11
4 MODELING WITH K-MEANS CLUSTERING	14
5 EVALUATION AND CONCLUSION.....	26
REFERENCES.....	28

List of Figures

Figure 1 - First Investigations of the Dataset	5
Figure 2 - One-Hot Encoding of Categorical Variables	6
Figure 3 - Maximum Values and Boxplot of the First Column "session"	7
Figure 4 - Maximum and Minimum Values of the Column "age"	8
Figure 5 - Boxplot for Column "age"	8
Figure 6 - Correlation Matrix for All Columns.....	9
Figure 7 - The First Five Entries of the Cleaned Dataset	12
Figure 8 - First 5 Entries and Shape of the NumPy Array X.....	13
Figure 9 - Initialization of the "Centroids" as Starting Points for the k-Means Algorithm	14
Figure 10 - Assignment of Points in the Dataset to Centroids and Output as an Index Array	15
Figure 11 - Calculation of the Updated Centroids.....	15
Figure 12 - k-Means Algorithm	16
Figure 13 - Initialization of k-Means for the Present Dataset	17
Figure 14 - Feature Distribution Across Segments 1	18
Figure 15 - Feature Distribution Across Segments 2.....	18
Figure 16 - Feature Distribution Across Segments 3.....	19
Figure 17 - Feature Distribution Across Segments 4.....	19
Figure 18 - Elbow Method for Scaled Dataset	21
Figure 19 - Elbow Method for Non-Scaled Dataset	21
Figure 20 - Implementation of Clustering with scikit-learn.....	22
Figure 21 - Scatter Plot Matrix for Clustering Results	23
Figure 22 - Excerpt from Clustering of the Scaled Dataset with 3 Clusters.....	24
Figure 23 - Excerpt from the Scatter Plot Matrix for the Clustering Results of the Unscaled Dataset.....	25

1 Business Understanding

For several years, the importance of data science in digital marketing has been steadily increasing. Due to the breadth of available data, machine learning can be used to identify patterns in the data that would otherwise remain undiscovered and, if applied correctly, can create additional value for the company (cf. Saura, 2021, p. 92f.). In the present marketing data science project for the online shop "sustainable-fit.com", the first step is to develop a business case to clearly define the business goals and understand how data science can contribute to achieving these goals. Understanding the business problems from a data-driven perspective can lead to better decision-making and prioritization of resources (cf. Provost & Fawcett, 2013, p. 52ff).

The team of the online shop is small and the shop itself is only a few months old. There are already many customers, but the turnover is to increase further by expanding the product range. The business case, therefore, consists of providing existing customers with a broader range of products according to their preferences, thus increasing cross- and up-selling (cf. Kubiak & Weichbroth, 2010, p. 3f.). In addition, providing customers with products that suit them would lead to stronger loyalty. The potential thus lies in the customer lifetime value, which is to be created and increased in the first place (cf. Borle et al., 2007, p. 100f.).

The use case that arises from this for marketing data science is the identification of the most promising customers who have the highest potential to become long-term customers. The interests and behavior of this segment are particularly crucial. From this, appropriate steps for expanding the product range could be derived.

As a success criterion, the customer lifetime value can be derived and measured as a KPI. This way, it can be checked to what extent the recommendations from the data science project actually benefit the business case. The measurement can be done in a simple way with a cost-benefit calculation or with prediction methods such as the Bayesian approach (cf. Borle et al., 2007, p. 100f.). This will not be discussed in more detail at this point, as a separate project would be necessary for this. It is important to mention that the GDPR and all other applicable laws must be complied with at all times (cf. Westerkamp, 2020). This not only serves the legal protection of the company but also the trust of the customers.

For the identification of the preferences and behavior of the most relevant customer group, segmentation by means of clustering is aimed at in order to gain valuable

insights into existing data. For this to succeed, some preparations must be made, which will be discussed in the following chapter.

2 Dataset & Data Cleaning

The careful selection and cleaning of the dataset is a critical step in any marketing data science project. This chapter is dedicated to the dataset and data cleaning for the 'sustainable-fit.com' project. Data cleaning provides the foundation to ensure that the decisions to be derived are based on reliable information (cf. Ridzuan & Wan Zainon, 2019, p. 731).

First, the dataset is loaded into a Pandas DataFrame (Pandas, 2024) to examine it more closely and then use it further. Initially, the columns were recognized as one, and in the third code block in the Jupyter Notebook (Jupyter, 2024), which can be seen in Figure 1, this problem was solved by separating the columns. In the fourth code block, the first relevant information about the dataset is generated, which can be seen in the output. This includes the number of entries, the number of unique entries, the column names, missing values, and the first rows. This allows developing an initial feel for the dataset.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

[2] ✓ 5.9s

df = pd.read_csv("/Users/johanncasparzahn/Desktop/python_projects/dataset_kundensegmentierung.csv")

[3] ✓ 0.0s

# Split the data into separate columns
df = df['session;newsletter_abo;checkout;addtocart;eingeloeste_rabatte;analytics_referer_type;produktkategorie_interesse;purchase;wishlist;geschlecht;alter;analytics_campaign_source'].str.split(';').expand=True

# Set the column names
df.columns = ['session', 'newsletter_abo', 'checkout', 'addtocart', 'eingeloeste_rabatte', 'analytics_referer_type', 'produktkategorie_interesse', 'purchase', 'wishlist', 'geschlecht', 'alter', 'analytics_campaign_source']

# Convert columns to appropriate data types
df[['session', 'newsletter_abo', 'checkout', 'addtocart', 'eingeloeste_rabatte', 'purchase', 'wishlist']] = df[['session', 'newsletter_abo', 'checkout', 'addtocart', 'eingeloeste_rabatte', 'purchase', 'wishlist']].apply(pd.to_numeric)

[34] ✓ 0.0s

# Informationen über den Datensatz
print(f"Einträge: {len(df)}")

num_unique_rows = df.drop_duplicates().shape[0]
print(f"Zahl der einzigartigen Einträge: {num_unique_rows}")

print("\nSpalten:")
for column in df.columns:
    print(f"  {column}")

# Informationen zu Datentypen und fehlenden Werten
print("\nDatentypen und fehlende Werte:")
info_df = pd.DataFrame(df.dtypes, columns=['Datentyp'])
info_df['Fehlende Werte'] = df.isnull().sum()
info_df['Eindeutige Werte'] = df.nunique()
print(info_df)

# Erste Zeilen des Datensatzes
print("\nErste Zeilen des Datensatzes:")
print(df.head())

[34] ✓ 0.0s

...
Einträge: 1111
Zahl der einzigartigen Einträge: 1035

Spalten:
- session
- newsletter_abo
- checkout
- addtocart
- eingeloeste_rabatte
- analytics_referer_type
- produktkategorie_interesse
- purchase
- wishlist
- geschlecht
- alter
- analytics_campaign_source

Datentypen und fehlende Werte:

```

	Datentyp	Fehlende Werte	Eindeutige Werte
session	int64	0	72
newsletter_abo	int64	0	2
checkout	int64	0	2
addtocart	int64	0	2
eingeloeste_rabatte	int64	0	2
analytics_referer_type	object	0	4

```
...
1  48  0  1  1  0  direct  Apfelsaft  0  0  weiblich  44  google
2  48  0  0  0  0  search  Waldsaft  0  0  weiblich  55  google
3  28  0  1  1  1  campaign  Waldsaft  1  0  weiblich  32  google
4  28  0  1  1  0  direct  Traubensaft  1  1  weiblich  46  dankesmail

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Figure 1 - First Investigations of the Dataset

In the next step, the categorical variables are converted into values that can be better used for statistical purposes later on. As shown in Figure 2, One-Hot-Encoding was used here to split the categorical column "analytics_referer_type" into four columns, each indicating whether the assignment to "campaign", "direct", "search", or "website" is true or false (cf. Zhu et al., 2024, p. 2). This was also done for the columns "produktkategorie_interesse" (product category interest), "geschlecht" (gender), and "analytics_campaign_source".

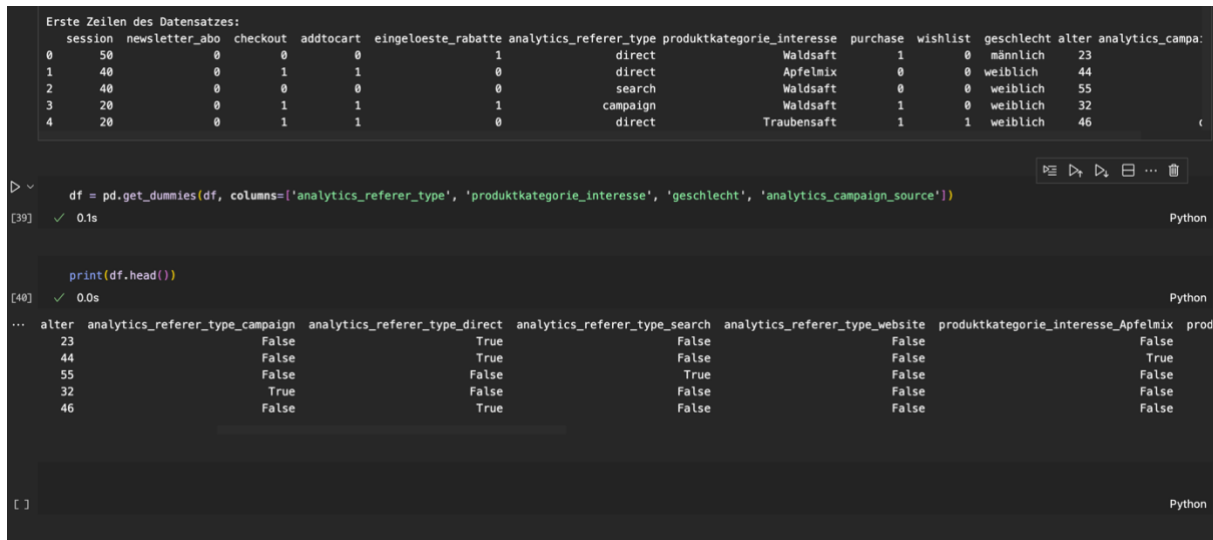


Figure 2 - One-Hot Encoding of Categorical Variables

Furthermore, it is important to identify outliers in the dataset that could negatively influence later modeling. The two columns that could be affected in this case are "session" and "age". The other columns are boolean values and therefore cannot "deviate" in a numerical sense. Figure 3 shows the analysis of the "session" column. Based on the properties, it is assumed that this refers to the number of sessions of the user. The output of the maximum values and the boxplot show (cf. Degen, 2010) that there are some outliers that go into the thousands and are not realistic. Therefore, all entries with a session count above 200 are removed from the dataset.

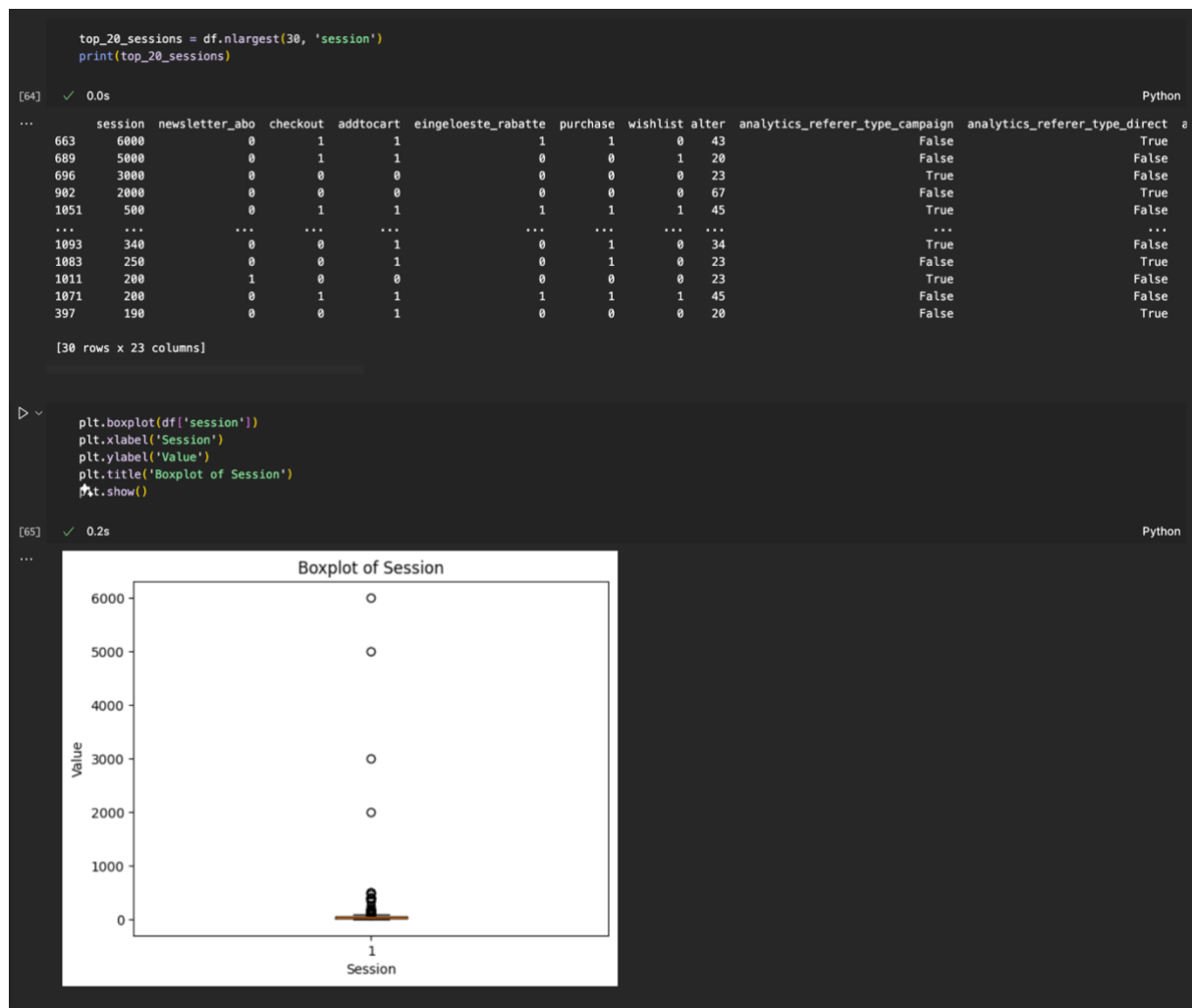


Figure 3 - Maximum Values and Boxplot of the First Column "session"

Figures 4 and 5 show the analyses of the age column. These indicate that there are no unnatural outliers. The maximum value is 88 years and then steadily decreases, which is realistic. The lowest value is 10 years, and as can be seen, this occurs 12 times. The age of 12 years is shown once, and then it continues from 18 years onwards. Since the data is about customers of an online shop, only persons of legal age are considered relevant for segmentation, and the entries with 10 and 12 years are cleaned up.


```
df['alter'] = df['alter'].astype(int)

top_20_alter = df.nlargest(20, 'alter')
print(top_20_alter)
```

[70] ✓ 0.0s Python

...	session	newsletter_abo	checkout	addtocart	eingeloeste_rabatte	purchase	wishlist	alter	analytics_referer_type_campaign	analytics_referer_type_direct
10	12	0	0	0	0	0	1	88	False	True
507	10	0	0	0	0	0	1	80	False	True
536	20	0	0	0	0	0	1	80	False	True
557	23	0	0	0	0	0	1	80	False	True
604	30	0	1	1	0	0	1	80	True	False
1038	20	0	0	1	0	1	0	80	True	False
1060	30	0	1	1	1	1	0	80	False	False
1087	340	0	1	1	1	1	0	80	False	False
1092	5	0	1	1	1	1	0	80	True	False
13	30	0	0	0	0	0	1	76	False	True
178	20	1	0	0	0	0	0	70	False	True
386	10	0	1	1	0	0	1	70	True	False
514	40	0	1	0	0	0	1	70	False	True
799	30	0	0	0	0	0	0	70	True	False
821	20	0	0	0	0	0	0	70	False	True
9	50	0	1	1	1	1	1	67	True	False
32	23	0	0	0	0	0	1	67	False	True
84	1	0	0	0	0	0	1	67	False	True
102	23	0	0	0	0	0	0	67	False	False
147	30	0	0	0	0	0	1	67	True	False


```
low_20_alter = df.nsmallest(20, 'alter')
print(low_20_alter)
```

[71] ✓ 0.0s Python

...	session	newsletter_abo	checkout	addtocart	eingeloeste_rabatte	purchase	wishlist	alter	analytics_referer_type_campaign	analytics_referer_type_direct
232	20	0	0	0	0	0	1	10	True	False
251	20	0	0	0	0	0	1	10	True	False
333	20	0	0	0	0	0	1	10	False	True
440	20	0	0	0	0	0	1	10	True	False
450	20	0	1	0	0	1	1	10	True	False
645	40	0	0	0	0	0	1	10	False	False
664	10	0	0	0	0	0	0	10	False	True
746	60	0	1	1	1	1	1	10	False	False
861	30	0	1	1	0	0	0	10	True	False
880	20	0	1	1	1	1	1	10	False	True
962	160	0	1	1	1	1	1	10	False	False
1080	10	0	1	1	1	1	0	10	False	True
16	10	0	0	0	0	0	1	12	False	False
6	23	0	1	1	0	0	1	18	False	True
26	30	0	1	1	1	1	1	18	False	True
61	50	0	1	0	0	1	0	18	True	False
78	30	0	0	1	0	0	0	18	True	False
89	45	0	0	0	0	0	1	18	False	False
96	60	0	0	0	0	0	0	18	False	True
132	20	0	0	0	0	0	1	18	False	True

Figure 4 - Maximum and Minimum Values of the Column "age"

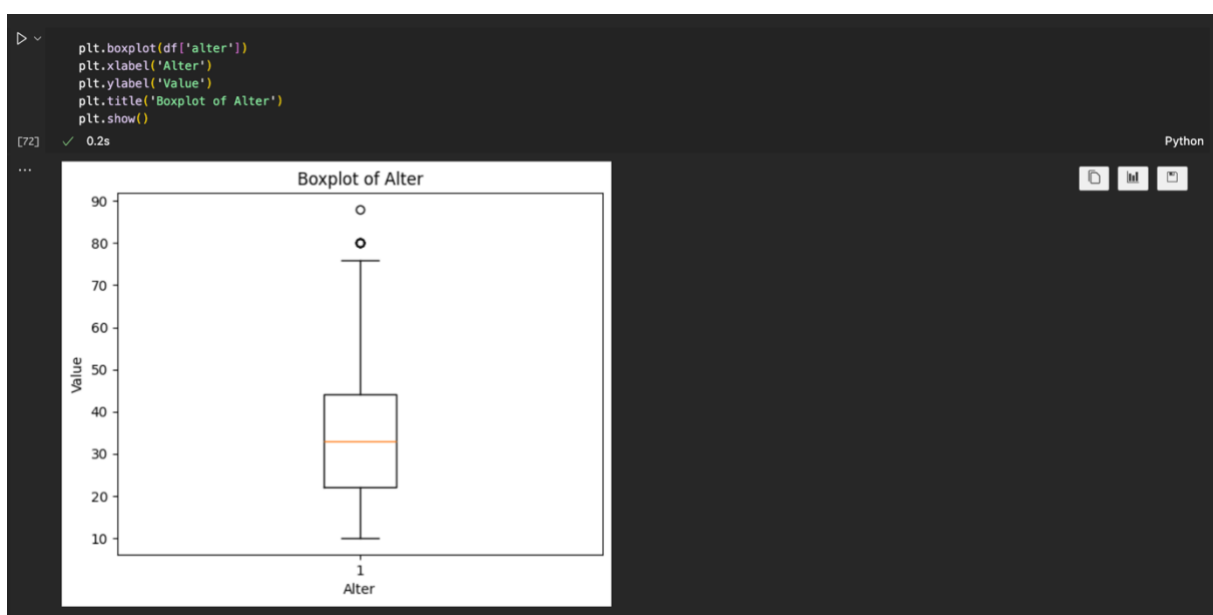


Figure 5 - Boxplot for Column "age"

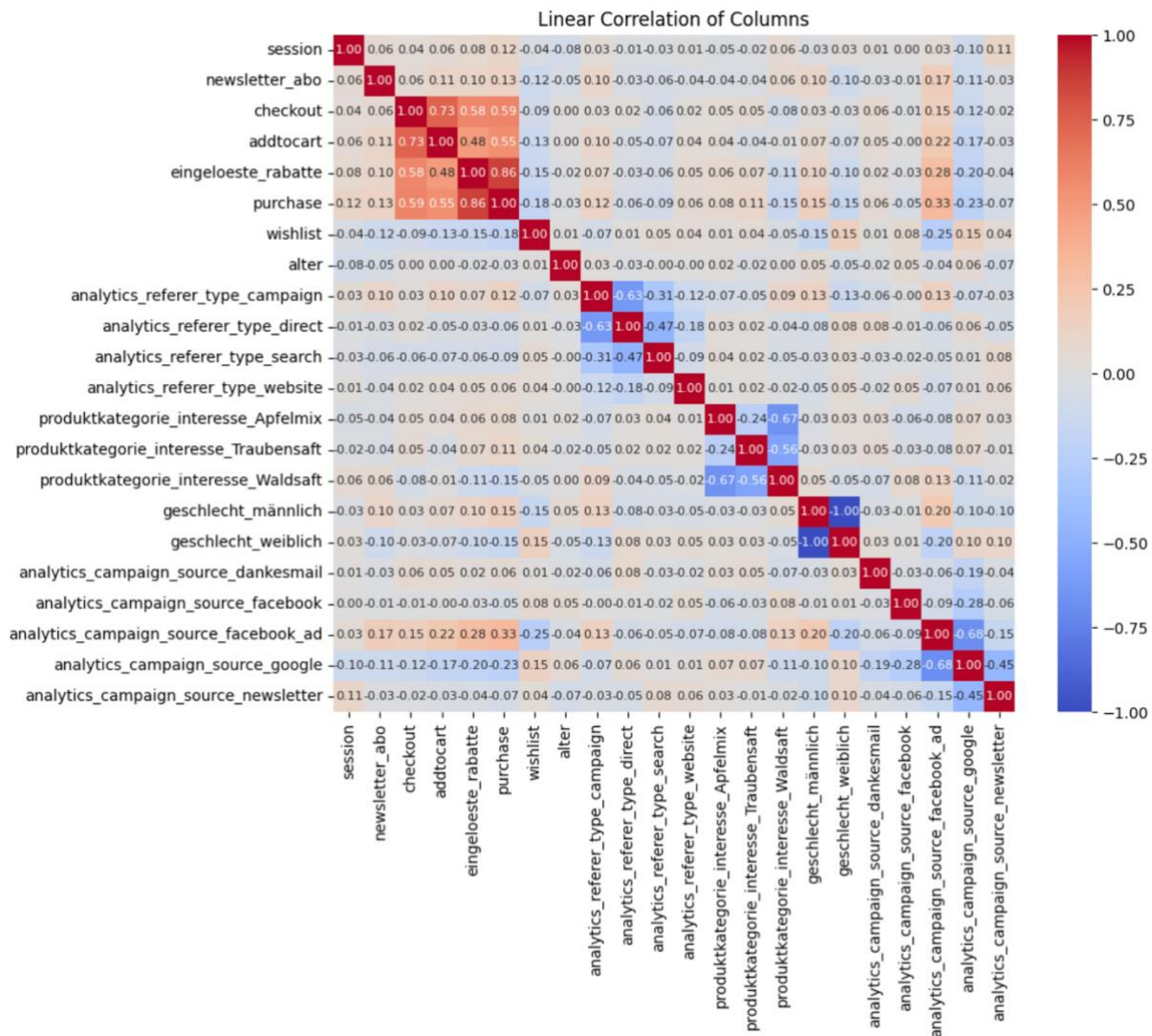


Figure 6 - Correlation Matrix for All Columns

The next step is to visualize the correlations of the different data fields. Figure 6 shows a correlation matrix (cf. Degen, 2010), which was generated for the present cleaned dataset using the Python packages "Seaborn" and "Matplotlib" (cf. Seaborn, 2024 & Matplotlib, 2024). The color of the fields makes the relationships easily recognizable. Blue fields indicate a negative correlation, red fields a positive correlation, and gray fields indicate little to no correlation.

Some areas stand out, such as the area in the upper left, which is characterized by strong positive correlations. This refers to the actions of users in the Lower Funnel (cf. Wiesel, Pauwels & Arts, 2011, p. 604ff), from adding a product to the shopping cart to making a purchase. Since many users who add a product to their cart or enter the checkout area also become buyers, the correlation of these actions is strongly positive.

Other areas that stand out are, for example, that Facebook Ads have a strong correlation of 0.33 with purchases, which suggests good optimization of these ads. However, men also show a positive correlation of 0.2 with Facebook Ads. Here, it could be examined whether ads for women could be optimized to better bring this target group into the funnel. The opposite is true for Google and newsletters as campaign sources.

3 Selection of Funnel Phase and Feature Selection

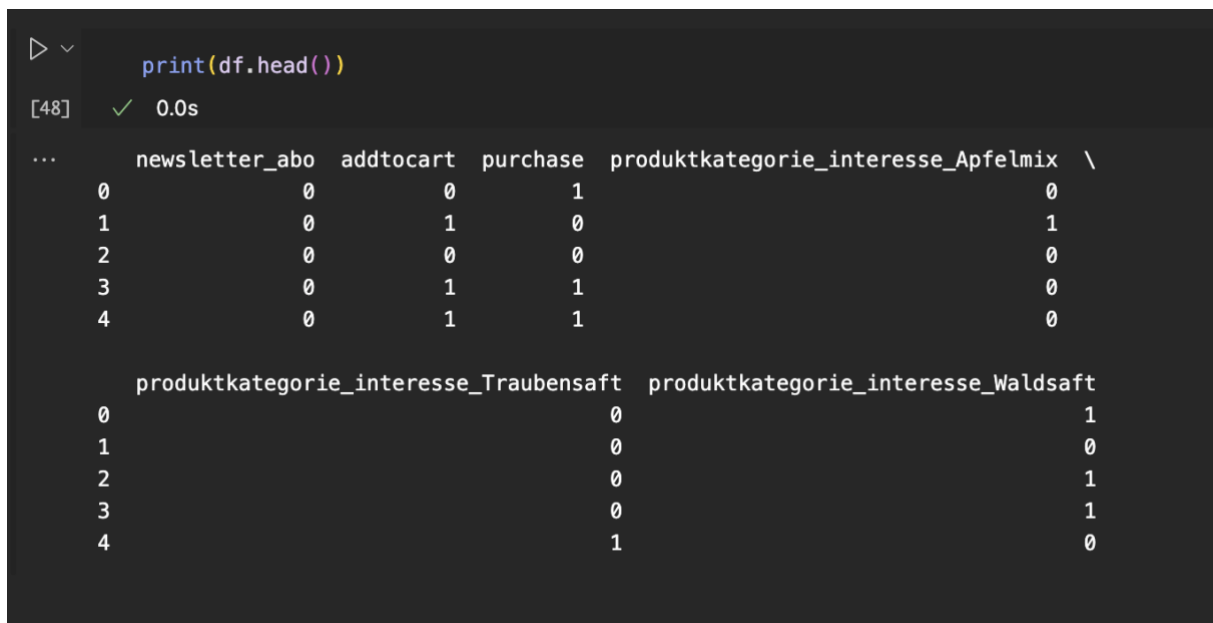
The correlation matrix will further help to extract the right features for the part of the funnel to be selected (cf. Hall, 1999). The initial situation of the online shop is that it has been online for some time and has good customer numbers, but there is potential in sales. The question of particular business interest is in which direction the product range should be further expanded. In the context of the online sales funnel (cf. Wiesel et al., 2011 & Szymkowiak, 2019), the phase that is most worthwhile to analyze must therefore be identified. The analysis of the upper funnel phase would provide information about which users access the online shop's site and how, and what steps they take. Relevant features for this would be sessions, age, gender, referer type, campaign source, or product category interest. However, in the context of the current business situation, the upper funnel is less interesting, as the number of potential customers is not the main problem.

More interesting for the expansion of the product range are the middle and lower funnel, where interest in specific products arises and this interest culminates in a purchase through the execution of several actions. Relevant features for the middle funnel are the wishlist tracked here and the product category interest can also be integrated into this phase. Crucial for the lower funnel are the features newsletter, add-to-cart, check-out, redeemed discounts, and purchase. Against the background of the use case, the relevant features will be selected in the next step.

Product category interest is essential to gain insights into which direction the product range should be expanded. It provides direct information about customer preferences. Including the newsletter feature is also useful, as it identifies customers with a sustained interest in the shop. The preferences of these customers should therefore be included in the modeling. Additionally, add-to-cart and purchase are important to include actual buying behavior from adding to the cart to completing the purchase, while considering possible deficits in the ordering process. Check-out is thus no longer necessary, as the feature correlates with add-to-cart at 0.73 and redundancy should be avoided. The same applies to redeemed discounts, which correlate with purchase at 0.86 and would not bring valuable added value to the modeling. The wishlist shows a negative correlation with purchase completion, therefore does not serve the business goal and is omitted. The features age and gender could bring insights, but also unnecessarily distort the model. The focus should be on purchase-related features and

not on demographic data. They are therefore also excluded, as are the sessions, which also do not contribute to the business case. This initial selection of features will be reflected in the evaluation of the model and adjusted if necessary.

Next, a normalization of the features would follow to balance the weighting for the modeling (cf. Muhammad Ali & Faraj, 2014). In this case, however, all selected features have binary values, i.e. 0 or 1. The categorical data has already been converted using binary decomposition (one-hot encoding). Figure 7 shows the first five entries of the cleaned dataset.



```
print(df.head())
```

[48] ✓ 0.0s

	newsletter_abo	addtocart	purchase	produktkategorie_interesse_Apfelmix	produktkategorie_interesse_Traubensaft	produktkategorie_interesse_Waldsaft
0	0	0	1	0	0	1
1	0	1	0	1	0	0
2	0	0	0	0	0	1
3	0	1	1	0	0	1
4	0	1	1	0	1	0

Figure 7 - The First Five Entries of the Cleaned Dataset

For using the data in modeling, it must be transferred into a NumPy array (cf. NumPy Developers, 2023). In this case, it was named "X", and Figure 8 shows the first five rows with the binary values for the six selected features and the so-called "shape", which represents the form of the matrix.

```
print(f"Die ersten fünf Reihen von X: \n {X[:5]}")  
print(f"Shape von X: {X.shape}")
```

[52] ✓ 0.0s

```
... Die ersten fünf Reihen von X:  
[[0 0 1 0 0 1]  
 [0 1 0 1 0 0]  
 [0 0 0 0 0 1]  
 [0 1 1 0 0 1]  
 [0 1 1 0 1 0]]  
Shape von X: (1071, 6)
```

Figure 8 - First 5 Entries and Shape of the NumPy Array X

4 Modeling with k-Means Clustering

For segmentation, the k-Means clustering algorithm is chosen (Deepali Virmani et al., 2015 & Su Chang, 2018 & Syakur et al., 2018). This algorithm is widely used and simple to implement. The algorithm consists of two repeating steps. First, as many arbitrary points in the 6-dimensional space are selected as clusters are to be created. These points are called "Centroids". Usually, points from the dataset are selected randomly for this purpose. The implementation of this initialization can be seen in Figure 9. In the first step, all 1071 points from the dataset are assigned to the nearest centroid. This process can be seen in Figure 10. The assignment is stored in the index array "idx". The second step consists of moving the centroids, figuratively speaking, to the center of their assigned points. To do this, each centroid is made the mean of all points assigned to it. This process can be seen in Figure 11. This is followed again by the first step. This process is repeated as many times as necessary to arrive at representative clusters; in this case, ten iterations were initially chosen. Figure 12 shows the summarized process, and Figure 13 shows the execution for the present dataset with 3 clusters.

```
def kMeans_init_centroids(X, K):  
    """  
    This function initializes K centroids that are to be  
    used in K-Means on the dataset X  
  
    Args:  
        X (ndarray): Data points  
        K (int):      number of centroids/clusters  
  
    Returns:  
        centroids (ndarray): Initialized centroids  
    """  
  
    # Randomly reorder the indices of examples  
    randidx = np.random.permutation(X.shape[0])  
  
    # Take the first K examples as centroids  
    centroids = X[randidx[:K]]  
  
    return centroids  
[54] ✓ 0.0s
```

Figure 9 - Initialization of the "Centroids" as Starting Points for the k-Means Algorithm

```
def find_closest_centroids(X, centroids):
    """
    Computes the centroid memberships for every example

    Args:
        X (ndarray): (m, n) Input values
        centroids (ndarray): (K, n) centroids

    Returns:
        idx (array_like): (m,) closest centroids

    """

    # Set K
    K = centroids.shape[0]

    idx = np.zeros(X.shape[0], dtype=int)

    for i in range(X.shape[0]):
        values = []
        for j in range(K):
            values.append(np.linalg.norm(X[i] - centroids[j]))
        idx[i] = np.argmin(values)

    return idx
```

Figure 10 - Assignment of Points in the Dataset to Centroids and Output as an Index Array

```
def compute_centroids(X, idx, K):
    """
    Returns the new centroids by computing the means of the
    data points assigned to each centroid.

    Args:
        X (ndarray): (m, n) Data points
        idx (ndarray): (m,) Array containing index of closest centroid for each
            example in X. Concretely, idx[i] contains the index of
            the centroid closest to example i
        K (int): number of centroids

    Returns:
        centroids (ndarray): (K, n) New centroids computed

    """

    m, n = X.shape

    centroids = np.zeros((K, n))

    for i in range(K):
        points = X[idx == i]
        centroids[i] = np.mean(points, axis = 0)

    return centroids
```

✓ 0.0s

Figure 11 - Calculation of the Updated Centroids


```

def run_kMeans(X, initial_centroids, max_iters=10):
    """
    Runs the K-Means algorithm on data matrix X, where each row of X
    is a single example
    """

    # Initialize values
    m, n = X.shape
    K = initial_centroids.shape[0]
    centroids = initial_centroids
    idx = np.zeros(m)
    plt.figure(figsize=(8, 6))

    # Run K-Means
    for i in range(max_iters):

        # Output progress
        print("K-Means iteration %d/%d" % (i, max_iters-1))

        # For each example in X, assign it to the closest centroid
        idx = find_closest_centroids(X, centroids)

        # Given the memberships, compute new centroids
        centroids = compute_centroids(X, idx, K)
    plt.show()
    return centroids, idx

```

[55] ✓ 0.0s

Figure 12 - k-Means Algorithm

```

for i in range(5):
    # Set number of centroids and max number of iterations
    K = 3
    max_iters = 10

    # Set initial centroids by picking random examples from the dataset
    initial_centroids = kMeans_init_centroids(X, K)

    # Run K-Means
    centroids, idx = run_kMeans(X, initial_centroids, max_iters)

    Number_of_segment_0 = np.sum(idx == 0)
    Number_of_segment_1 = np.sum(idx == 1)
    Number_of_segment_2 = np.sum(idx == 2)
    print(f"Number of customers in segment 0: {Number_of_segment_0}")
    print(f"Number of customers in segment 1: {Number_of_segment_1}")
    print(f"Number of customers in segment 2: {Number_of_segment_2}")

    segment_0 = np.zeros((Number_of_segment_0, X.shape[1]))
    segment_1 = np.zeros((Number_of_segment_1, X.shape[1]))
    segment_2 = np.zeros((Number_of_segment_2, X.shape[1]))

    j_0 = 0
    j_1 = 0
    j_2 = 0

    for i in range(len(X)):
        if idx[i] == 0:
            segment_0[j_0] = X[i]
            j_0 += 1
        elif idx[i] == 1:
            segment_1[j_1] = X[i]
            j_1 += 1
        elif idx[i] == 2:
            segment_2[j_2] = X[i]
            j_2 += 1

    # Calculate the mean values of features for each segment
    segment_means = {}
    for segment, data in zip(['Segment 0', 'Segment 1', 'Segment 2'], [segment_0, segment_1, segment_2]):
        if len(data) > 0:
            segment_means[segment] = np.mean(data, axis=0)
        else:
            segment_means[segment] = np.zeros(X.shape[1])

    # Create a dataframe with the mean values
    segment_means_df = pd.DataFrame(segment_means, index=df.columns)

    # Plot stacked bar chart
    ax = segment_means_df.plot(kind='bar', stacked=True, figsize=(10, 6))
    ax.set_xlabel('Features')
    ax.set_ylabel('Mean')
    ax.set_title('Feature Verteilung über die Segmente')
    ax.legend(title='Segmente')

    # Add labels for each segment
    for i in range(len(segment_means_df)):
        total = 0
        for j in range(len(segment_means_df.columns)):
            value = segment_means_df.iloc[i, j]
            if value > 0:
                ax.text(i, total + value/2, f'{value:.2f}', ha='center', va='center')
            total += value

    plt.tight_layout()
    plt.show()

```

Figure 13 - Initialization of k-Means for the Present Dataset

Figure 14-17 shows the visualized results of this first clustering attempt. Each figure shows the result of clustering with different starting points. Since only binary properties

are present here, i.e., whether something is the case or not, a bar chart was chosen for visualization, in which the colors represent the segments and show how large the arithmetic mean of the segment is for each category.

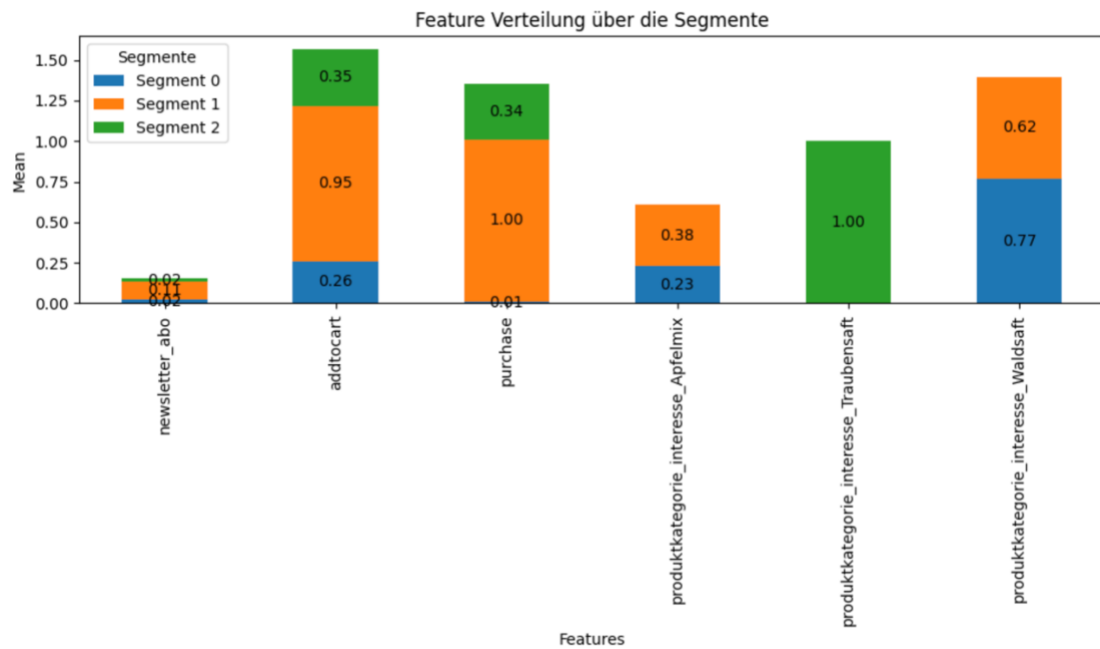


Figure 14 - Feature Distribution Across Segments 1

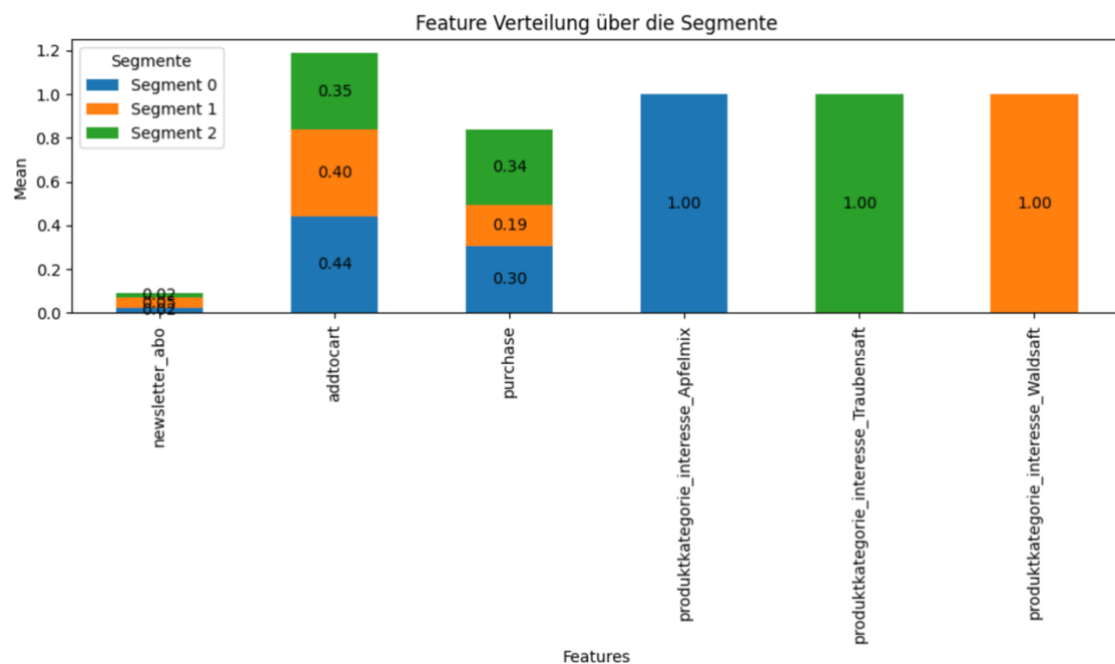


Figure 15 - Feature Distribution Across Segments 2

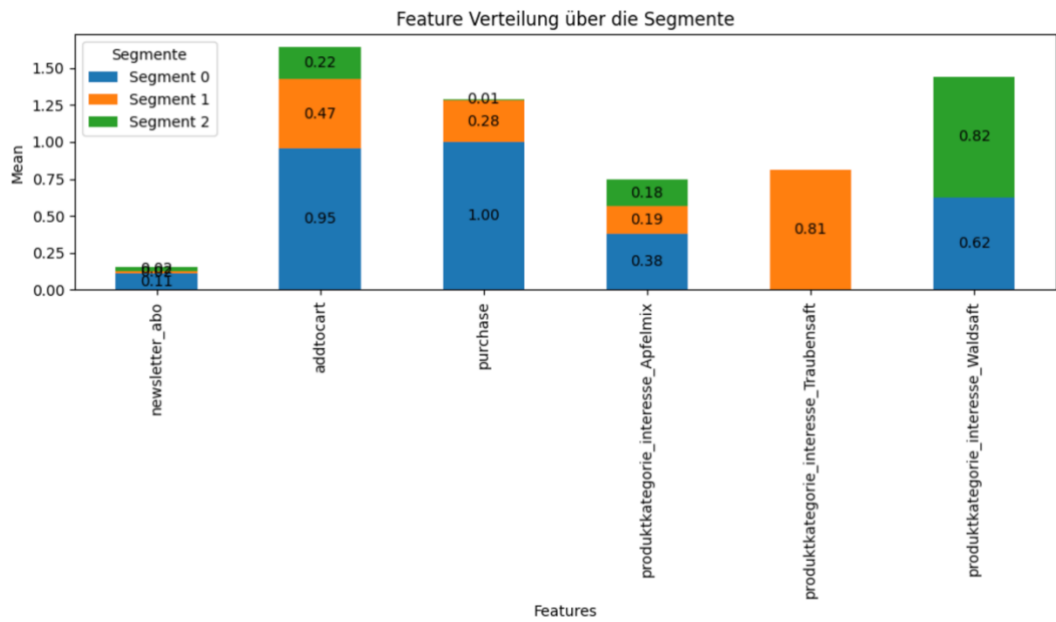


Figure 16 - Feature Distribution Across Segments 3

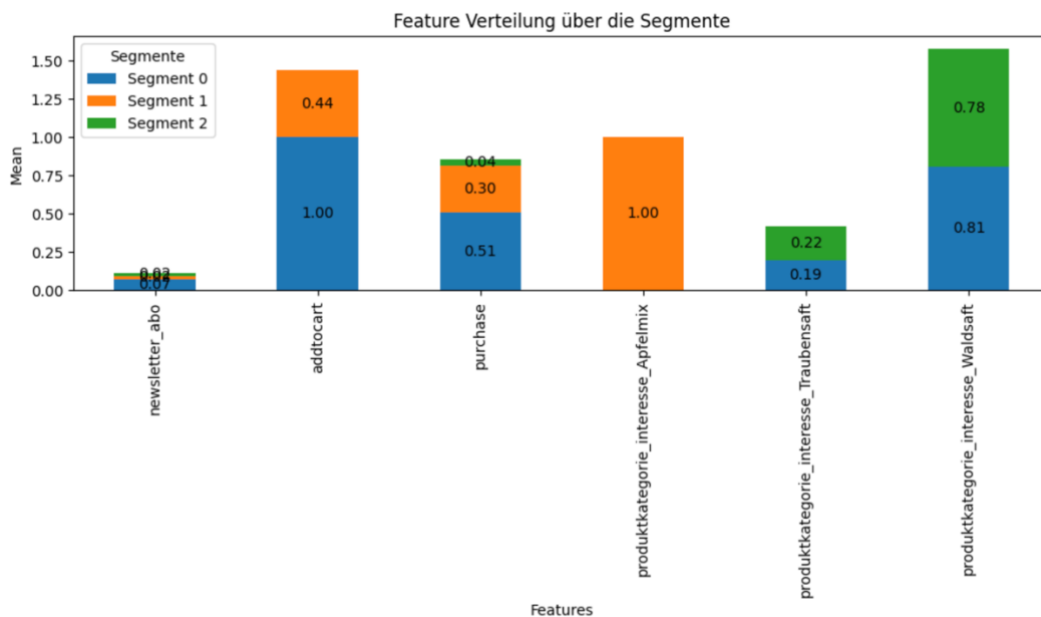


Figure 17 - Feature Distribution Across Segments 4

At first glance, the four iterations of clustering appear highly inconsistent. However, upon closer examination, content-related conclusions can already be drawn, which are evident in each of the four diagrams. The orange segment 1 in Figure 14 can serve as an illustration. The algorithm has filled this cluster exclusively with buyers (which can be seen from the "Mean" of 1.00 for "purchase"). This group of buyers is primarily interested in Waldsaft and, to a lesser extent, in Apfelmix, and the comparatively high

number of newsletter subscriptions suggests a strong connection to this group. While Figure 15 is less informative because the segmentation merely resembles filtering by product interest, Figures 16 and 17 also show the strength of Waldsaft enthusiasts in purchases and newsletter subscriptions.

Despite these initial insightful results, the clustering will be performed again using a different strategy. By incorporating the features of age and gender, more insight into the people behind the segments is to be gained. In addition, the product category interest will be weighted with the number of sessions to pursue two goals. First, converting binary to continuous values is intended to enhance the algorithm's performance. Second, product interest can thus be linked to the amount of interactions, adding more information to the product interest. An argument against weighting is that, for example, users who have high product interest but never have many interactions on websites due to their online behavior will be neglected because their value will be smaller. Therefore, two algorithms will be tested, one with the weighting of product interest and one without.

For gender, after converting into two binary variables (male yes/no and female yes/no), only one variable is used because a user is male if they are not female and vice versa. Age is scaled to values between 0 and 1 using the `MinMaxScaler` from `scikit-learn` (cf. Scikit-learn Developers, 2024).

Moreover, in this second iteration of the clustering model, the complete implementation in Python is omitted, and instead, the `scikit-learn` tool is used to abstract the implementation to a simpler level (cf. Scikit-learn Developers, 2024). Additionally, the optimal number of clusters is determined this time using the "Elbow Method" (cf. Syakur et al., 2018, pp. 3f). In this method, the number of clusters is varied (here between 1 and 10), and for each number of clusters, the clustering algorithm is applied, and at the end, the sum of all squared distances of each data point to its cluster center is determined. The lower the sum, the more accurate the clusters are. However, they should also not be too precise, as in the extreme case, a cluster would be created for each data point. A graph of the number of clusters against the sum of distances helps to find the point that gives the method its name. There, the sum drops sharply before and significantly less afterward. Figures 18 and 19 show the graphs for the dataset scaled with the sessions and the unscaled dataset.

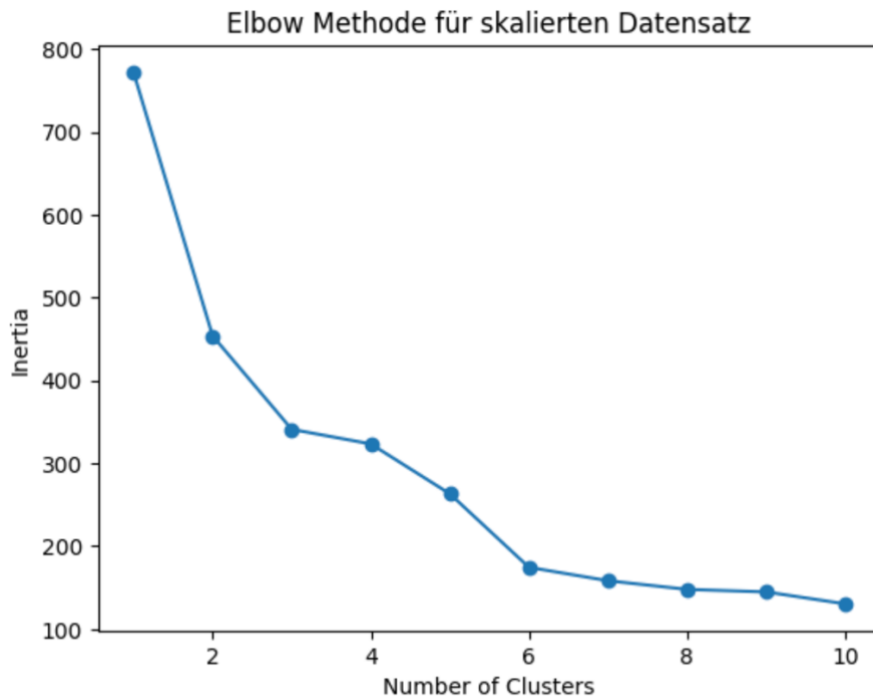


Figure 18 - Elbow Method for Scaled Dataset

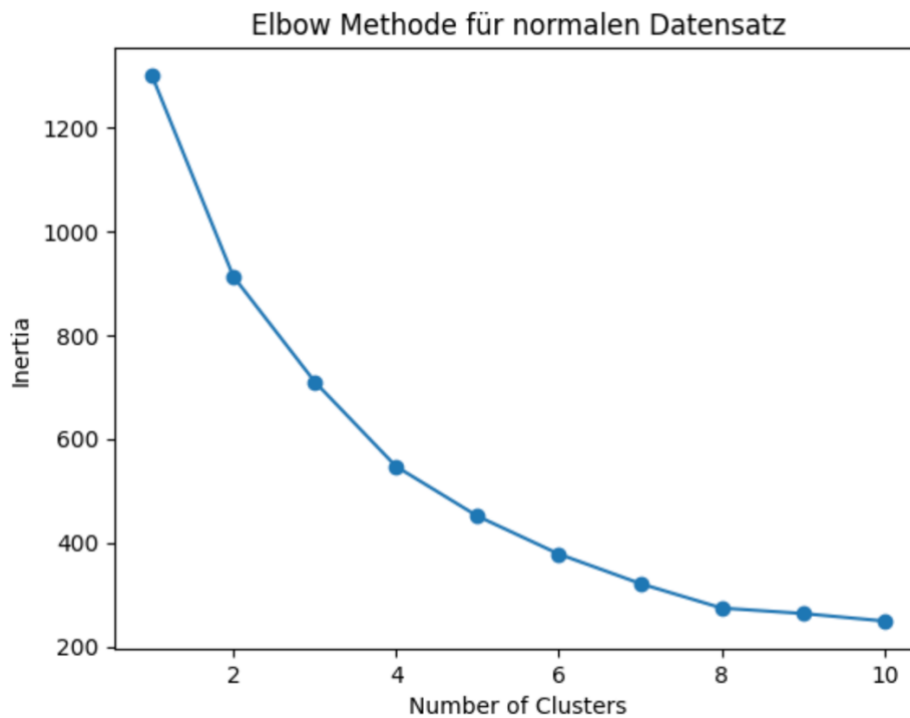


Figure 19 - Elbow Method for Non-Scaled Dataset

The graphs do not show a clear picture, however, for the scaled dataset, a cluster count of 3 or 6 seems to be the best choice, and for the unscaled dataset, upon closer examination, 4 can be considered a reasonable decision. In the next step, the

clustering is implemented for these parameters, and the results are visualized. The code for this can be seen in Figure 20.

```
# Fit the KMeans algorithm to the data
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(df_scaled)

# Get the cluster labels for each data point
cluster_labels = kmeans.predict(df_scaled)

# Add the cluster labels as a new column in the dataframe
df_scaled['Cluster'] = cluster_labels

# Visualize the clusters using a scatter plot matrix
sns.pairplot(df_scaled, hue='Cluster', palette='viridis')
plt.show()
```

Figure 20 - Implementation of Clustering with scikit-learn

For visualization, all variables are plotted against each other in a matrix, and the data points are colored according to their cluster. Figure 21 shows the entire matrix for the scaled dataset with 3 clusters.



Figure 21 - Scatter Plot Matrix for Clustering Results

For the scaled dataset, in which product interest was weighted by the number of sessions, an interesting observation can be made quickly. In the implementation with three clusters, the algorithm, similar to the first iteration, filled one cluster with the buyers. This can be seen in Figure 22. The top row of graphs shows the variable Purchase (with 1 or "yes" at the top and 0 or "no" at the bottom) against various other variables. The upper points are all yellow, thus part of Cluster 2. The scatter plot in the bottom right corner is interesting. There, age is plotted against interest in products from the "Waldsaft" (forest juice) category. In this case, age acts as a good distribution of points, making it clear that the buyers, which are colored yellow, are mainly on the left side of the graph, while many turquoise and also violet points and only a few yellow

ones occur on the right side. Since product interest was weighted by the number of sessions, this graph indicates that a higher number of sessions does not indicate a higher probability of a purchase and thus not more interest, at least not in the sense that would be commercially interesting. Thus, it can be concluded that the weighting was not conducive to the model as originally intended. However, the attempt has given us valuable information, namely that buyers who are interested in Waldsaft require very few sessions before making the purchase. For the other two product categories, it can be seen that the yellow points are more widely distributed, even more so than the non-buyers from the other clusters. Thus, the Waldsaft category seems to lead to purchases the fastest, which again argues for the expansion of this product category.

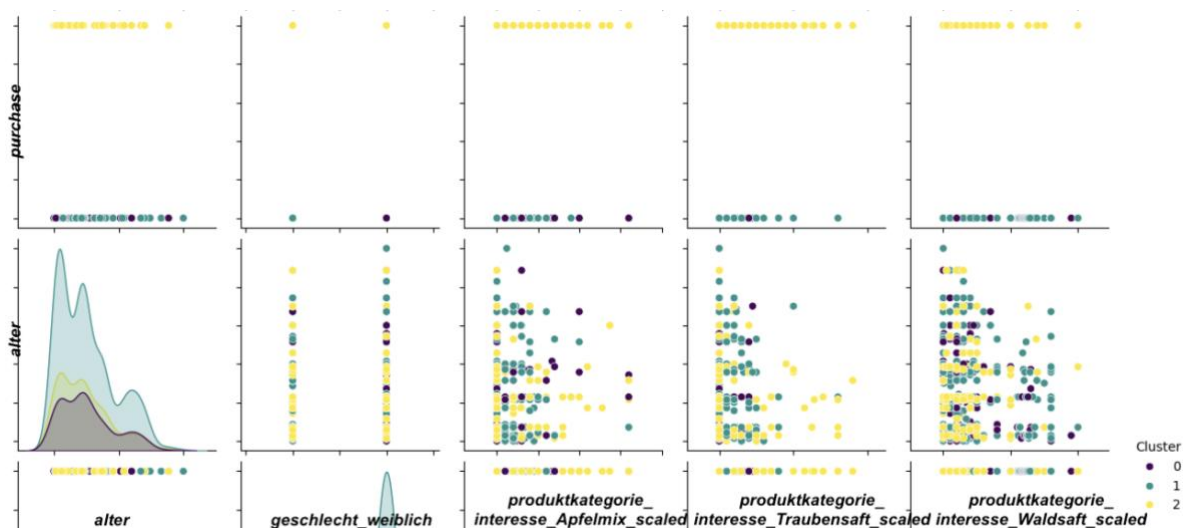


Figure 22 - Excerpt from Clustering of the Scaled Dataset with 3 Clusters

The second clustering of the scaled dataset with six clusters turns out to be too confusing, especially in the visualization of the matrix used here. No added value or new information could be obtained from the majority of the clusters. The clustering with the unscaled dataset can also offer little additional information. The strong influence of the binary values for product category interest led, as seen in Figure 23, to a segmentation by interest, here the randomly matching assignment of yellow for apple mix, purple for grape juice, and green and blue for forest juice. In any case, it can be seen that age and gender should not play a role as decisive or relevant variables in the decision about business expansion. Age is very evenly distributed across all product interests, as are the genders.

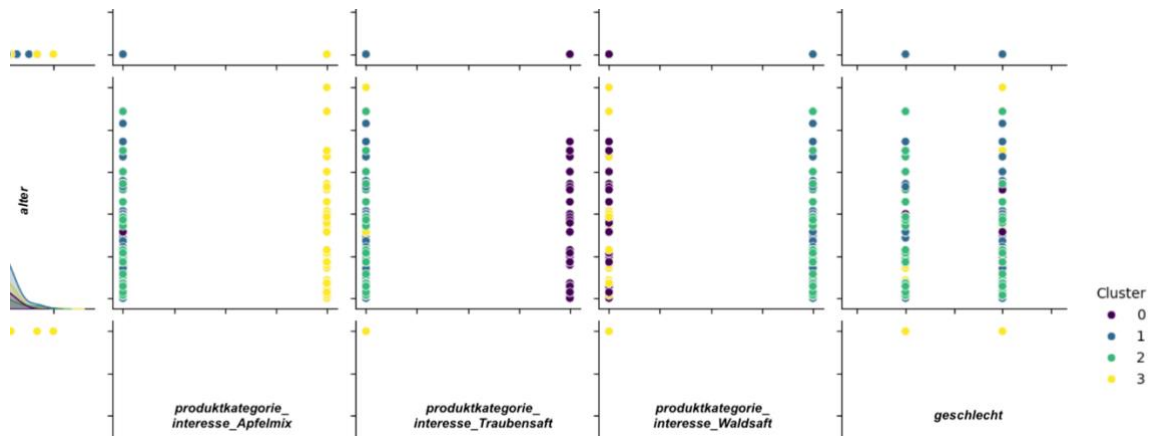


Figure 23 - Excerpt from the Scatter Plot Matrix for the Clustering Results of the Unscaled Dataset

5 Evaluation and Conclusion

The evaluation of the results of the clustering model compared to the situation before the implementation shows significant improvements in terms of customer segmentation and the resulting opportunities for targeted business strategies. Before the application, there was no systematic segmentation of the customer base of the online shop 'sustainable-fit.com'. However, by using the k-means algorithm, important insights could be gained.

The model quality can be considered satisfactory, as the clusters show clear differences in the selected features and thus allow meaningful conclusions to be drawn (Stein & Vollnhals, 2011, p. 5). In particular, the identification of a cluster of buyers who show a strong interest in the product category "forest juice" and quickly proceed to purchase provides valuable insights for the alignment of the product range and the development of targeted marketing measures.

Nevertheless, there are also limitations of the clustering model to consider. The use of mainly binary variables limits the possibilities of differentiation between the clusters. Moreover, weighting the product interest with the number of sessions did not lead to the expected results despite insightful hints, as a higher number of sessions does not necessarily correlate with a higher purchase probability. For future iterations of the model, the inclusion of additional continuous variables could be considered. For this purpose, the existing tracking tools should be checked for possibilities and possibly data from the ERP (Enterprise Resource Planning) should be added (cf. Gupta & Kohli, 2006, p. 687).

In order to use the results of the clustering model in practice, the segment of customers interested in forest juice should be included, especially against the background of expanding the product range. The optimization of the online shop and the adaptation of the product range can contribute to increasing sales and improving customer loyalty.

For a successful deployment of the clustering model, it is important to involve the relevant stakeholders in the company and create acceptance for the project. This includes communicating the results and potential benefits for the company to management and affected employees. Training and workshops can help familiarize employees with the new approaches and facilitate their implementation in day-to-day business. Regular feedback and continuous monitoring of results are also crucial to adapt and optimize the model as needed. The data situation regarding customer

behavior can always change, which is why modeling should take place at regular intervals with up-to-date data. For this purpose, an automation script can be implemented in the cloud, for example, which performs the clustering weekly and sends the visualized results to management (cf. Amazon Web Services, Inc., 2024).

Overall, the developed clustering model provides a solid foundation for customer segmentation and optimization of the business strategy of the online shop 'sustainable-fit.com'. Through continuous development and refinement of the model as well as close cooperation with relevant stakeholders, the potential of marketing data science for the company can be fully exploited.

References

- Amazon Web Services, Inc. (2024). AWS Lambda – Serverless Compute. Available at: <https://aws.amazon.com/lambda/> [Accessed on: 31.03.2024].
- Borle, S., Singh, S.S. & Jain, D.C. (2007). Customer Lifetime Value Measurement. *Management Science*, 54(1), 100-112. <https://doi.org/10.1287/mnsc.1070.0746>
- Degen, H. (2010). Graphische Datenexploration. In *Handbuch der sozialwissenschaftlichen Datenanalyse*. Springer.
- Degen, H. (2010). Graphische Datenexploration. In: Wolf, C., Best, H. (Hrsg.), *Handbuch der sozialwissenschaftlichen Datenanalyse* (S. 91-116). VS Verlag für Sozialwissenschaften. https://doi.org/10.1007/978-3-531-92038-2_5
- Gupta, M. & Kohli, A. (2006). Enterprise resource planning systems and its implications for operations function. *Technovation*, 26(5–6), 687-696. <https://doi.org/10.1016/j.technovation.2004.10.005>
- Hall, M.A. (1999). Correlation-based Feature Selection for Machine Learning. Dissertation, The University of Waikato, Hamilton, New Zealand.
- Jupyter (2024). Project Jupyter - Open-source software, open standards, and services for interactive computing across dozens of programming languages. Available at: <https://jupyter.org/> [Accessed on: 31.03.2024].
- Kubiak, B. & Weichbroth, P. (2010). Cross- And Up-selling Techniques In E-Commerce Activities. *Journal of Internet Banking and Commerce*, 15.
- Matplotlib (2024). Matplotlib: A plotting library for the Python programming language and its numerical mathematics extension NumPy. Available at: <https://matplotlib.org/> [Accessed on: 31.03.2024].
- Muhammad Ali, P.J. & Faraj, R.H. (2014). Data Normalization and Standardization: A Technical Report. *Machine Learning Technical Reports*, 1(1), 1-6.
- NumPy Developers (2023). NumPy: The fundamental package for scientific computing with Python. Available at: <https://numpy.org/> [Accessed on: 31.03.2024].
- Pandas (2024). Python Data Analysis Library. Available at: <https://pandas.pydata.org/> [Accessed on: 31.03.2024].

Provost, F. & Fawcett, T. (2013). Data Science and its Relationship to Big Data and Data-Driven Decision Making. *Big Data*, 1(1), 51-59.

Ridzuan, F. & Wan Zainon, W.M.N. (2019). A Review on Data Cleansing Methods for Big Data. *Procedia Computer Science*, 161, 731-738.
<https://doi.org/10.1016/j.procs.2019.11.177>

Saura, J. R. (2021). Using Data Sciences in Digital Marketing: Framework, methods, and performance metrics. *Journal of Innovation & Knowledge*, 6(2), 92-102.

Scikit-learn Developers (2024). scikit-learn: Machine Learning in Python. Available at: <https://scikit-learn.org/stable/> [Accessed on: 31.03.2024].

Seaborn (2024). Seaborn: statistical data visualization. Available at: <https://seaborn.pydata.org/> [Accessed on: 31.03.2024].

Stein, P. & Vollnhals, S. (2011). Grundlagen clusteranalytischer Verfahren. Institut für Soziologie, Universität Duisburg-Essen.

Su Chang, Xu Zhenzong, Gao Xuan, 2018. Improvement of K Mean Clustering Algorithm Based on Density. <http://arxiv.org/abs/1810.04559v1>

Szymkowiak, A. (2019). Marketing in Online Sales Funnels. *Preuzeto*, 5, 2021.

Syakur, M.A., Khotimah, B.K., Rochman, E.M.S. & Satoto, B.D. (2018). Integration K-Means Clustering Method and Elbow Method For Identification of The Best Customer Profile Cluster. In: *IOP Conference Series: Materials Science and Engineering*, Volume 336, The 2nd International Conference on Vocational Education and Electrical Engineering (ICVEE), 9 November 2017, Surabaya, Indonesia. IOP Publishing. 012017. <https://doi.org/10.1088/1757-899X/336/1/012017>

Westerkamp, C. (2020). Datenschutz gemäß DSGVO im datengetriebenen Marketing – ein Überblick. In Boßow-Thies, S., Hofmann-Stölting, C., Jochims, H. (Hrsg.), *Data-driven Marketing* (S. 237-256). Springer Gabler, Wiesbaden.
https://doi.org/10.1007/978-3-658-29995-8_11

Wiesel, T., Pauwels, K. & Arts, J. (2011). Practice Prize Paper—Marketing's Profit Impact: Quantifying Online and Off-line Funnel Progression. *Marketing Science*, 30(4), 604-611.

Zhu, W., Qiu, R. & Fu, Y. (2024). Comparative Study on the Performance of Categorical Variable Encoders in Classification and Regression Tasks. <https://doi.org/10.48550/arXiv.2401.09682>