Projektarbeit

für das Modul

Grundlagen Sozialer Netze

an der

Hochschule für Technik und Wirtschaft (HTW) Berlin Fachbereich 4: Informatik, Kommunikation und Wirtschaft, Studiengang Angewandte Informatik

Gutachterin: Prof. Dr. Christin Schmidt

Eingereicht von: Max Hager

Datum: 12.07.2021

Inhaltsverzeichnis

Vorwort

Die vorliegende Arbeit ist eine schriftliche Unterstützung zu einer Präsentation für das Modul Grundlagen Sozialer Netze. Ziel soll es sein, einen umfangreichen Überblick über das in dem Modul behandelte Thema in Form eines Projektes zu geben. Dabei handelt es sich um eine Projekt mit allen darin beinhalteten Punkten aus dem Kreislauf der Datenanalyse. Durch die Anweisungen und der Bereitstellung der jeweiligen Materialien für das Studienfach Grundlagen Sozialer Netze, geleitet von Prof. Dr. Christin Schmidt, war es der Gruppe möglich, semesterbegleitend diese Arbeit fertig zu stellen.
Anhand des Einflusses von COVID-19 auf die Musikwelt ist es möglich gewesen, den gesamten Prozess der Datenanalyse abzubilden.
Die folgenden genannten Personen sehen von sämtlichen Datenschutzrichtlinien ab und sind mit freier Verwendung der Informationen der Arbeit einverstanden:

- Nathanael Isaac Hermanto - 570619 (NIH)
- Kevin Kessel - 568289 (KK)
- Edward Rau - 576405 (ER)
- Max Gordon Brüschke - 575397 (MGB)
- Max Hager - 575522 (MH)

1 Einleitung

Musik ist bekannt dafür, dass sie die jeweilige Stimmung in der Gesellschaft widerspiegelt - das hat schon Martin Scherber, ein deutscher Komponist, erkannt: „Die Musik soll alles befruchten. In ihr sind die Weltgeschehnisse wohl auf kürzeste Art zusammengefaßt.“[1]. Diese Aussage gestaltet den maßgeblichen Inhalt der vorliegenden Arbeit, in welcher die Folgen von COVID-19 auf die musikalische Abbildung der Weltgeschehnisse - nach Scherber - analysiert werden.

1.1 Problemformulierung

Dadurch, dass es offensichtlich Zusammenhänge zwischen Gesellschaft und Musik gibt, gilt damit die Frage, ob dies auch im Kontext zu Covid zutrifft. Daher wird folgende Forschungsfrage untersucht:

“Wie wirkt sich Corona und die damit einhergehende Schließung von Clubs/Festivals auf die Tanzbarkeit und die Positivität von elektronischer Musik aus?”

2 Grundlage und Theorie

Wie bereits in der Einleitung erläutert, gibt es immer wieder Zusammenhänge zwischen der Musik und der Gesellschaft zu den jeweiligen Zeitpunkten in der Historie [2]. Die Melodie von Musik, also die Abfolge von Tönen in Takten, kann nachweisbar unseren Gefühlszustand beeinflussen [3]. Ein Ton entsteht, wenn Gegenstände, wie zum Beispiel zwei Hände, sich anfangen zu bewegen. Durch die Bewegung werden die sich in der Luft oder in Wasser befindlichen Partikel in Schwingung gesetzt und es entsteht eine Vibration, von der wir - durch das Gehör in unserem Gehirn angekommen - von einem Ton sprechen. Je nachdem, wie schnell ein Gegenstand die Partikel entweder wegdrückt oder heranzieht, entsteht eine Frequenz. Diese kann man beispielsweise mit Hilfe einer Welle darstellen, bei der man die Zyklen durch das Weg - oder Heranziehen der Partikel durch den Gegenstand darstellt [Abbildung 1]. Die Zyklen werden Frequenz (Hertz) genannt - die Frequenz errechnet sich daraus, wie viele Zyklen es pro Sekunde gibt. Niederfrequente, aufeinander prallende Partikel, bezeichnen wir als Bässe und hochfrequente als Höhen. In der Musik wird demnach nichts anderes getan, als verschiedene Frequenzen aneinander zu reihen, die im Gesamtbild eine musikalisch attraktive Melodie erzeugen. Ist es das Ziel, Menschen zum Tanzen zu bringen und angenommen, man wisse, welche subjektiven Bedürfnisse die Zielgruppe hat, wäre eine gute Melodie also eine Zusammensetzung der Frequenzen, die genau dieses subjektive Ziel erfüllt. Es gibt zwar in der Musiktheorie allgemein anerkannte Töne, die beispielsweise mit Positivität assoziiert werden [4]. Allerdings ist die Frequenz der aufeinander prallenden Partikel ebenso eine subjektive Wahrnehmung eines jeden einzelnen Menschen.

Abbildung 1: Bewegung Partikel

Aufgrund dieser Tatsache sollte es möglich sein, anhand von Melodien, wie es bei elektronischer Musik üblich ist, die Stimmung eines Menschen zu beeinflussen. Diese Form der Musik enthält einerseits Melodien, die eine positive Stimmung anregen sollen und andererseits Melodien, die diese Tanzbarkeit und Munterkeit unterdrücken sollen und damit einhergehend das genaue Gegenteil ersterer Intentionen verursacht. Dieser Grundgedanke hat die Überlegung angeregt, ob durch die derzeitige Situation mit COVID-19 eine Veränderung in der Musik wahrnehmbar wird, weil die Gesellschaft sich scheinbar in einer Position befindet, in der sich die allgemeine Stimmung, beziehungsweise die Lebensstandards, geändert haben. Diese Änderungen sollen auf Basis von Musiktracks erkannt werden und die Mutmaßung, dass Musik gesellschaftliche Entwicklungen abbildet, bestärken.

3 Methodologie und Untersuchungsplanung

Für die folgende Arbeit vorgesehen sind sowohl eine Regressionsanalyse als auch ein Clustering. Die Regressionsanalyse soll die Fragestellung prüfen, inwiefern sich die Schließung von Klubs und das Absagen von Festivals und Konzerten auf neu erschienene Musik ausgewirkt hat. Dabei wird eine Liste mit insgesamt 100 Berliner DJs betrachtet [5]. Diese wurde stichprobenartig darauf geprüft, ob die Künstler in Berlin wohnhaft sind. Zur Untersuchung der Musik werden sogenannte Features betrachtet - Kennzahlen, welche aus einer Analyse der verwendeten Spotify API gewonnen werden. Von den von Spotify bereitgestellten Features werden im Folgenden zwei untersucht. Die Danceability eines Songs basiert auf dem Tempo, Rhythmus und Beat des Songs und repräsentiert die Tanzbarkeit eines Liedes. Das Feature Valence umfasst darüber hinaus die Tonhöhe und stellt dar, wie lebhaft ein Lied ist [6]. Benannte Features wurden aufgrund der Vermutung ausgewählt, dass sich bei diesen Features die größten Veränderungen abzeichnen. Vor der Regressionsanalyse soll ein Clustering durchgeführt werden. Dieses soll zeigen, dass nicht alle 100 DJs die gleichen Werte besitzen, was das Ergebnis der Regressionsanalyse verfälschen könnte. Für die Erhebung der Werte wird ein Python Wrapper verwendet [7].

4 Untersuchungsdurchführung

Die Regressionsanalyse ist in zwei Zeiträume unterteilt. Der erste Zeitraum reicht vom 01.01.2018 bis zum ersten Lockdown am 22.03.2020 und der zweite Zeitraum vom 22.03.2020 bis zum 23.06.2021. Dadurch können eventuelle Veränderungen vor und während der Corona-Pandemie besser nachvollzogen werden. Für die Regressionsanalyse werden drei Arrays benötigt. Ein Array beinhaltet jedes Datum im vorgegebenen Zeitraum. Die beiden verbleibenden Arrays müssen den Durchschnittswert eines Features - nach dem Datum sortiert - beinhalten. Für das Clustering wird der Durchschnittswert eines Features für alle Lieder des Künstlers benötigt.
Die Spotify API bietet eine Methode, um alle Werte für alle Features eines Liedes zu ermitteln. Die Werte liegen zwischen 0.0 und 1.0, wobei 1.0 sehr tanzbar bzw. lebhaft ist. Außerdem werden die Werte bis auf mehr als zehn Nachkommastellen berechnet, wodurch diese sehr genau sind. Für wenige Lieder gibt es keine Werte. Dies kommt allerdings so selten vor, dass es keine Auswirkung auf die Untersuchung hat. Das Ergebnis der Analyse ist ein Paket mit allen anderen ermittelten Werten nach der Standard Python Objekt Hierarchie. Somit müssen die benötigten Werte herausgefiltert werden. Zu beachten ist, dass ein Künstler verschiedene Versionen eines Liedes herausbringt, z.B. Extended Versions, Radio Mixes oder Remixes. Damit die Werte sich nicht doppeln, müssen diese aussortiert werden, um das Ergebnis der Regressionsanalyse nicht zu verzerren. Ebenfalls zu beachten ist, dass nicht an jedem Tag im Zeitraum Lieder erschienen sind, wodurch Lücken mit Nullwerten entstehen.
Für die Erstellung der drei Arrays für die Regressionsanalyse wurde ein Python Programm geschrieben. Das Programm bekommt eine Liste mit URLs von Künstlern und einen zu untersuchenden Zeitraum und liefert am Ende die drei benötigten Arrays.
Am Anfang des Programms werden zwei zweidimensionale Arrays erstellt, in denen alle Werte gesammelt werden. Zudem werden zwei eindimensionale Arrays definiert, in denen die Durchschnittswerte gespeichert werden. Alle vier Arrays sind genau so lang wie das Array mit den Tagen als Einträgen, damit die Werte zuordenbar sind. Darüber hinaus werden wichtige Objekte und Variablen erstellt und definiert.
Das Programm durchläuft die Liste für jeden Künstler einzeln. Bei einem Durchlauf werden alle Alben des Künstlers auf deren Erscheinungsdatum geprüft. Wenn dieses in den zu untersuchenden Zeitraum passt, werden alle Lieder des Albums untersucht.

```python
# Alle Artists im Array Artists durchlaufen
for artist in Artists:
    # Alle Alben eines Artists ermitteln
    albumResults = spotifyObject.artist_albums(artist)
    albumResults = albumResults['items']
    artist_pack = spotifyObject.artist(artist)
    artist_name = artist_pack['name']
    print(artist_name)

    # Alle Alben des momentanen Artists durchlaufen
    for album in albumResults:
        # Prüfen, ob das Album in den Zeotraum passt
        releasedate = parse(album['release_date'])
        if (releasedate < end and releasedate > start):
            albumID = album['id']
            trackResults = spotifyObject.album_tracks(albumID)
            trackResults = trackResults['items']

            # alle Songs eines passenden Albums durchlaufen
            for item in trackResults:
```

Abbildung 2: Prüfen des Erscheinungsdatums

Bevor die Werte für die Features ermittelt werden, wird geprüft, ob der Song eine Extended Version oder Ähnliches ist. Dabei wird der Titel des Liedes in Einzelteile zerlegt, welche auf bestimmte Stichwörter geprüft werden. Die hier verwendete Methodik ist nicht perfekt, da die Namen nicht optimal zerlegt werden. Bei einem Testdurchlauf, wobei die Namen aller Songs ausgegeben werden, die zu untersuchen sind, sind nur noch sehr selten Extended Versions und Ähnliches zu finden, weshalb diese Methodik hier als hinreichend erscheint.

```python
# alle Songs eines passenden Albums durchlaufen
for item in trackResults:
    # Überprüfen, ob der Song ein Remix oder ähnliches ist
    original_song = 1
    track_name = item['name']
    track_name_sliced = track_name.split()
    for x in track_name_sliced:
        if (x == "Extended" or x == "(Extended)" or
            x == "Extended)" or x == "[Extended]" or
            x == "Mix" or x == "(Mix)" or x == "Mix)" or
            x == "[Mix]" or x == "Version" or x == "(Version)" or
            x == "Version)" or x == "[Version]" or
            x == "Edit" or x == "(Edit)" or x == "Edit)" or
            x == "[Edit]" or x == "Rework" or x == "(Rework)" or
            x == "Rework)" or x == "[Rework]" or x == "Remix" or
            x == "(Remix)" or x == "Remix)" or x == "[Remix]" or x == "Remix]" or
            x == "Session" or x == "(Session)" or x == "Session)" or
            x == "[Session]" or x == "Sessions" or
            x == "(Sessions)" or x == "Sessions)" or
            x == "[Sessions]" or x == "Mixed" or x == "Remastered" or
            x == "Live" or x == "(Mixed)"):
            original_song = 0

    # Wenn der Song kein Remix ist, wird der Wert im Collected Array abgespeichert
    if original_song == 1:
```

Abbildung 3: Prüfen auf Original

Wenn das Lied als Original identifiziert wird, wird mithilfe des von Spotipy bereitgestellten Befehls die Audio Analyse durchgeführt. Wenn keine Werte vorhanden sind, wird zum nächsten Lied gesprungen. Die Werte werden basierend auf dem Erscheinungsdatum des Albums in das zweidimensionale Array eingetragen und der nächste Song wird untersucht.

```
# Features ermitteln
analyse = spotifyObject.audio_features(trackURIs)
if analyse[0] != None:
    # Features filtern und Position im Array bestimmen
    analyse_dancability = analyse[0]['danceability']
    analyse_valence = analyse[0]['valence']
    datestamp = releasedate-start
    datestamp = datestamp.days

    # Einfügen der Werte
    if Collected_Danceability[datestamp][0] == 0:
        Collected_Danceability[datestamp][0] = analyse_dancability
    else:
        Collected_Danceability[datestamp].append(analyse_dancability)

    if Collected_Valence[datestamp][0] == 0:
        Collected_Valence[datestamp][0] = analyse_valence
    else:
        Collected_Valence[datestamp].append(analyse_valence)
```

Abbildung 4: Werte ermitteln

Am Schluss werden die Werte in den Arrays des zweidimensionalen Arrays zusammengerechnet und durch deren Anzahl geteilt, um den Durchschnittswert zu erhalten. Dieser wird dann im finalen Array gespeichert.

```
# Ermitteln des Durchschnitts eines Tages
n = 0
for row in Collected_Danceability:
    i = 0
    x = 0.0
    for ele in row:
        x = x+ele
        i = i+1
    result = x/i
    danceability[n] = result
    n = n+1

n = 0
for row in Collected_Valence:
    i = 0
    x = 0.0
    for ele in row:
        x = x+ele
        i = i+1
    result = x/i
    valence[n] = result
    n = n+1
```

Abbildung 5: Durchschnitt berechnen

Nun hat man die benötigten Arrays für die Regressionsanalyse und kann mit dieser fortfahren.

5 Clustering

5.1 Methodologie

Eine Clusteranalyse ist ein Verfahren, um Ähnlichkeiten in Datenbeständen zu entdecken [12]. Die entstehenden Gruppen werden Cluster genannt und deren Inhalt besitzt weitesgehend ähnliche Merkmale. Eine Clusteranalyse hat das Ziel, Gruppen in Daten zu identifizieren und zu analysieren [13].

Die verwendete Methode wird als k-Means Clustering bezeichnet - eine Methode, bei der große Datenmengen schnell verarbeitet werden können[14]. Hierfür werden zunächst die Datensätze in k Gruppen geteilt, wobei das „k“ die Anzahl der Cluster darstellt. Voraussetzung zur erfolgreichen Verwendung ist ein numerischer Datensatz, um eine Berechnung der Distanz der Datenpunkte durchzuführen [15].

Zur Bestimmung der passenden Anzahl an Clustern wird das Elbow-Kriterium verwendet. Innerhalb eines Diagramms, bei welchem die x-Achse die Anzahl der Cluster und die y-Achse die Summe der quadrierten Abweichungen abbildet, werden die jeweiligen Punkte zur Clustermitte eingetragen. Falls ein Knick erkennbar sein sollte, wird der Punkt als optimale Anzahl an Clustern gewählt. Da ab einem weiteren Punkt die Summe der quadrierten Abweichung sich nur noch leicht verändert, sinkt die Aussagekraft der einzelnen Cluster [16].

5.2 Durchführung

5.2.1 Datenbereinigung

Zunächst werden die Namen von 100 DJs aus Berlin bestimmt. Zum Erhalt der jeweiligen Features wird die Konsole der Spotify Web API verwendet[17].

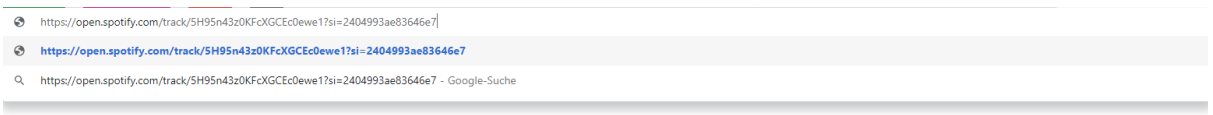

Abbildung 15: Songlink

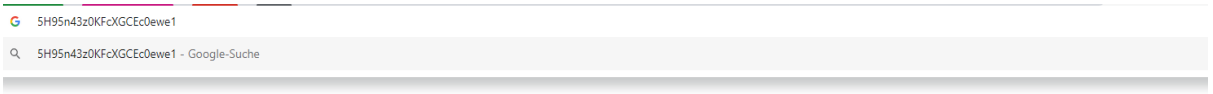

Abbildung 16: Song-ID

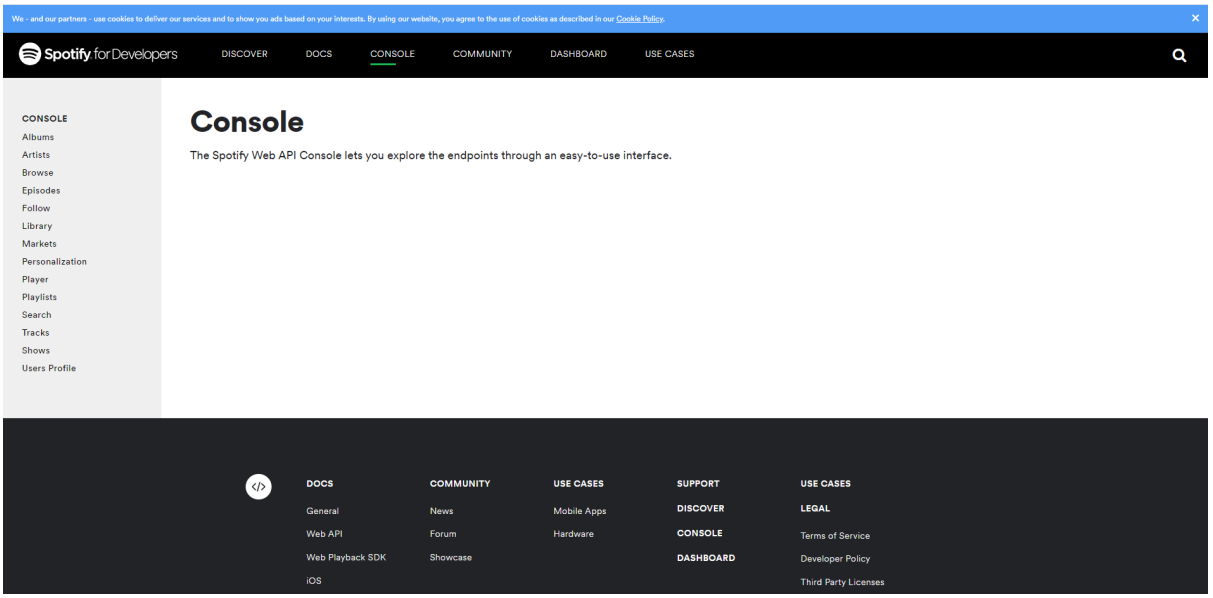

Abbildung 17: Konsole der Spotify Web API

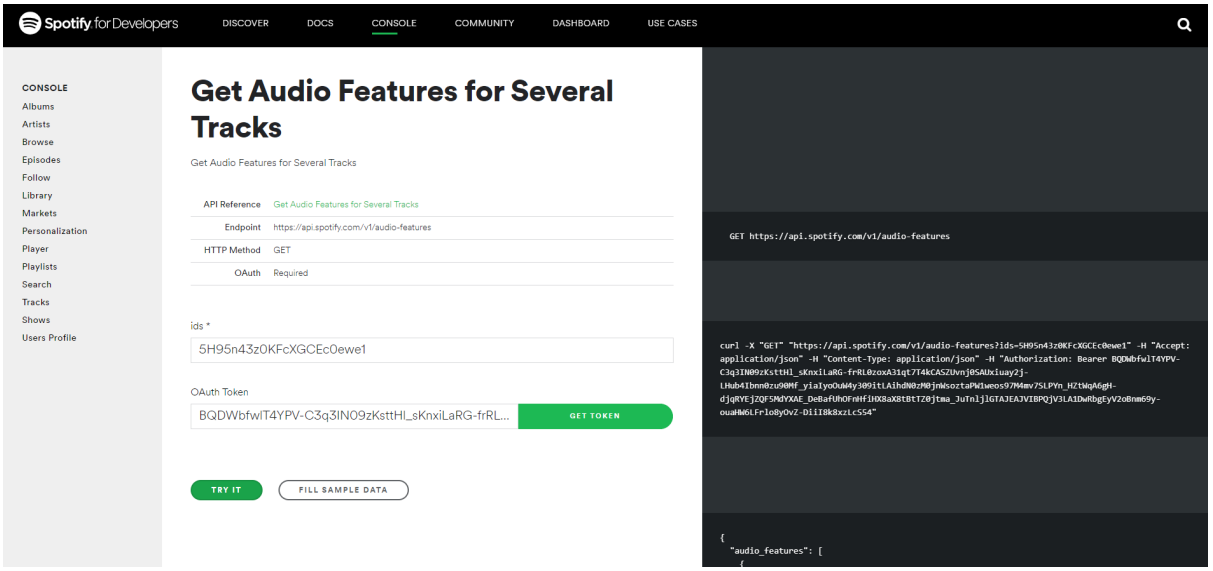

Abbildung 18: Einfügen der Song-ID in die Konsole

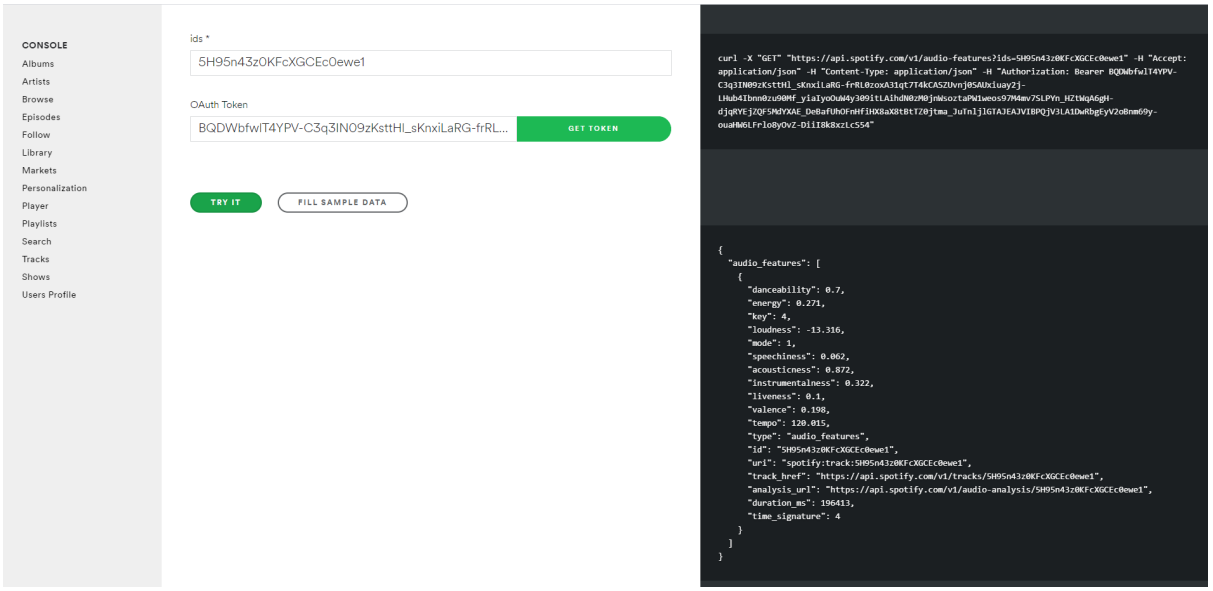

Abbildung 19: Ausgabe der Features, berücksichtigt werden Danceability und Valence

Die ID der Musiktitel wird herauskopiert und in die Konsole der Spotify Web API eingefügt, wodurch Werte hinsichtlich der Danceability und der Valence generiert werden.

Werke, die keinen direkten Zusammenhang zur Musik des Künstlers aufweisen - z. B. Remixe, die nicht vom jeweiligen Künstler selbst erstellt worden sind bzw. Podcasts des jeweiligen Künstlers - werden nicht berücksichtigt. Im Anschluss daran wird für jeden untersuchten Künstler ein Durchschnittswert der betrachteten Features berechnet, woraus sich eine Gesamtzahl von 200 Werten ergibt.

5.2.2 Modellerstellung

Auf Basis der bereinigten Datensätze wird ein Modell für das Clustering erstellt. Die untersuchten Features der Künstler, Danceability und Valence, werden verwendet um ein Modell für 2-dimensionales Clustering bei welchem die x-Achse die Danceability und die y-Achse die Valence(des jeweiligen Künstlers) illustriert. Bevor das Clustering modelliert wird, muss zunächst die optimale Anzahl an Clustern bestimmt werden. Hierbei wird die Elbow-Methode verwendet.

Python Quellcode für die Elbow-Methode

```
from sklearn.cluster import KMeans

from yellowbrick.cluster import KElbowVisualizer

import pandas as pd
```

Zuerst werden die benötigten Bibliotheken importiert.

```
df = pd.read_csv('artist_danceability_valence1.CSV')

model = KMeans()
```

Dann wird die Datei mittels Pandas als Data-Frame geladen und ein Modell mithilfe von KMeans erzeugt.

```
visualizer = KElbowVisualizer(model, k=(2,10), timings= True)

visualizer.fit(df[['Danceability', 'Valence']])

visualizer.show()
```

Die Funktion KElbowVisualizer passt das K-Means-Modell mit der Anzahl von Clustern im Bereich von 2 bis 9 an. Anschließend wird das Data-Frame im Visualizer angeglichen. Mittels Funktion show() kann der Visualizer auf dem Bildschirm angezeigt werden.

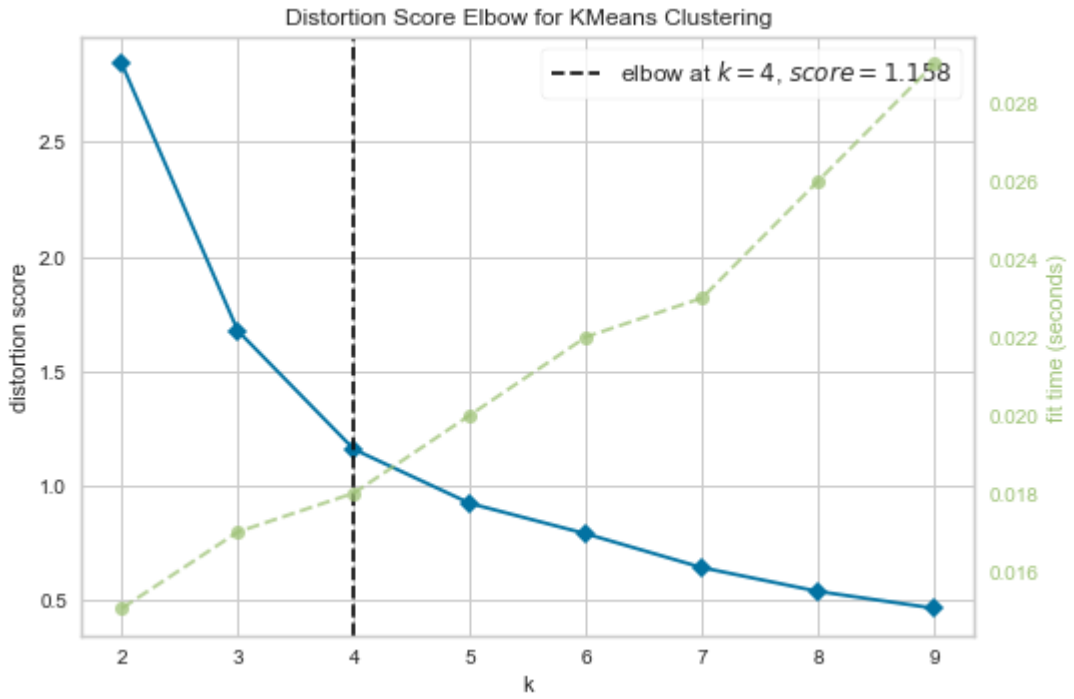

Abbildung 20: Grafik für Elbow Methode

Aus der oben dargestellten Grafik lässt sich schließen, dass die optimale Anzahl von Clustern 4 ist. Die Funktion informiert ebenfalls darüber, wie viel Zeit für das Zeichnen von Modellen für verschiedene Anzahlen von Clustern durch die grüne Linie benötigt worden ist.

Python Quellcode für Clustering

```
from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

from matplotlib.lines import Line2D

import pandas as pd

from scipy.spatial import ConvexHull

import numpy as np
```

Zuerst werden die benötigten Bibliotheken importiert.

```
df = pd.read_csv('artist_danceability_valence1.CSV')
```

Zum Öffnen, Lesen und zur Verwaltung der Datei wird Pandas verwendet.

```
kmeans = KMeans(n_clusters=4, random_state=0)
```

Danach wird das Clustering mit dem K-Means-Algorithmus eingestellt und die Anzahl an Clustern auf vier gesetzt.

```
df['cluster'] = kmeans.fit_predict(df[['Danceability', 'Valence']])
```

Mithilfe der Methode fit_predict wird ermittelt, welche Datei zu welchem Cluster gehören. Anschließend wird eine neue Spalte namens “cluster” in das Data Frame eingefügt.

```
centroids = kmeans.cluster_centers_

cen_x = [i[0] for i in centroids]

cen_y = [i[1] for i in centroids]


df['cen_x'] = df.cluster.map({0:cen_x[0],1:cen_x[1],2:cen_x[2], 3:cen_x[3]})

df['cen_y'] = df.cluster.map({0:cen_y[0],1:cen_y[1],2:cen_y[2], 3:cen_y[3]})
```

Anschließend werden die x- und y-Koordinaten für jeden Centroid vom jeweiligen Cluster bestimmt und in das Data Frame eingefügt.

```
colors = ['#DF2020', '#81DF20', '#2095DF', 'y']
df['c'] = df.cluster.map({0:colors[0],1:colors[1],2:colors[2], 3:colors[3]})
```

Mithilfe der Methode map wird jeder Stichprobe eine Farbe zugewiesen. Die Farben dienen dazu, die Stichproben unterscheiden zu können. Anschließend wird eine Spalte für die Farben in das Data Frame eingefügt.

```
#plotting
fig, ax = plt.subplots(1, figsize=(8,8))
# plot data
plt.scatter(df.Danceability, df.Valence, c=df.c, alpha=0.6, s=10)
# plot center
plt.scatter(cen_x, cen_y, marker="^", c=colors, s=70)
#draw enclosure
for i in df.cluster.unique():
        points = df[df.cluster == i][['Danceability', 'Valence']].values
        #get convex hull
        hull = ConvexHull(points)
        # get x and y coordinates
        # repeat last point to close the polygon
        x_hull = np.append(points[hull.vertices,0],
                points[hull.vertices,0][0])
        y_hull = np.append(points[hull.vertices,1],
                points[hull.vertices,1][0])
        # plot shape
```

```
        plt.fill(x_hull, y_hull, alpha=0.3, c=colors[i])


plt.xlim(0,1)

plt.ylim(0,1)
```

Schlussendlich können die Cluster visualisiert werden. Die Methode “subplots” erzeugt eine Figur und einen Satz von Unterplots (subplots). Dieser Utility-Wrapper ermöglicht gemeinsame Layouts von Unterplots einschließlich des umschließenden Figure-Objekts in einem einzigen Aufruf zu erstellen. Anschließend werden die Daten basierend auf den Variablen Danceability und Valence sowie die Centroide jeden Clusters mit Symbolen geplottet.

Um eine besser Visualisierung zu ermöglichen, wird Convex Hull verwendet. Die konvexe Hülle ist die kleinste Menge von Verbindungen zwischen den Datenpunkten, die ein Polygon bilden, das alle Punkte einschließt [18].

```
#legend elements

legend_elements = [Line2D([0], [0], marker='o', color='w', label='Cluster {}'.format(i+1), markerfacecolor=mcolor, markersize=5) for i, mcolor in enumerate(colors)]


#show legend elements

plt.legend(handles=legend_elements, loc='upper right')


#titles and labels

plt.title('Artists features \n', loc='left', fontsize=22)

plt.xlabel('Danceability')

plt.ylabel('Valence')
```

Unter Verwendung dieser Methode kann die Legende in der Visualisierung angezeigt werden. Bezeichnungen der x- und y-Achse werden in der Visualisierung sichtbar gemacht.

```
cluster1 = df[kmeans.labels_==0]

cluster2 = df[kmeans.labels_==1]

cluster3 = df[kmeans.labels_==2]

cluster4 = df[kmeans.labels_==3]
```

Um zu zeigen, welcher Künstler zu welchem Cluster gehört, werden zunächst neue Data-Frames für jeden Cluster erstellt.

```
cluster1.to_excel("cluster1.xlsx")

cluster2.to_excel("cluster2.xlsx")

cluster3.to_excel("cluster3.xlsx")

cluster4.to_excel("cluster4.xlsx")
```

Anschließend werden die Data Frames nach Excel exportiert, sodass die Künstler von jedem Cluster analysiert werden können.

Die Visualisierung von Clustering mittels des K-Means-Algorithmus.

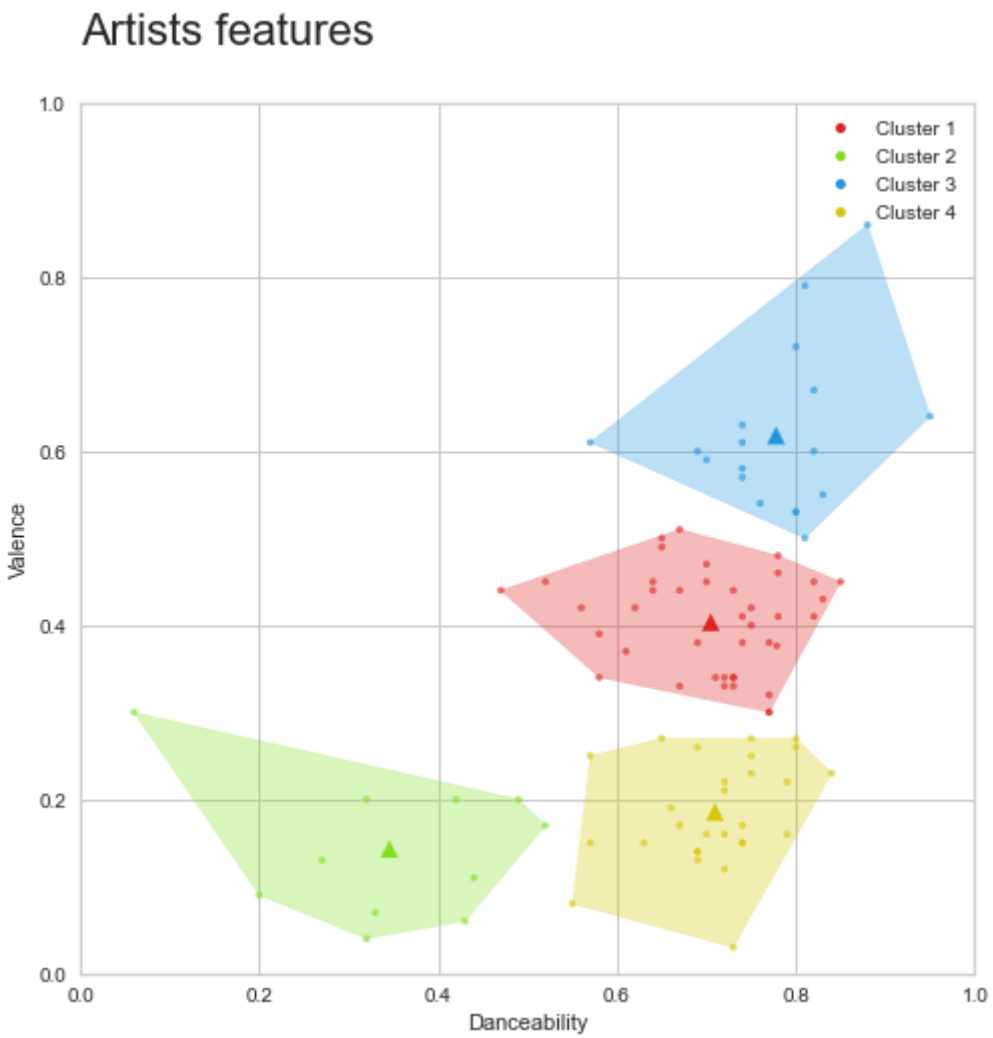

Abbildung 21: Die Visualisierung des Clusterings

Die Künstler im ersten Cluster:

Artist	**Danceability**	**Valence**	**cluster**	**cen_x**	**cen_y**	**c**
Paul Kalkbrenner	0.73	0.44	0	0.704713	0.403156	#DF2020
Alle Farben	0.7	0.47	w0	0.704713	0.403156	#DF2020
Thomas Gold	0.64	0.44	0	0.704713	0.403156	#DF2020
Acid Pauli	0.75	0.4	0	0.704713	0.403156	#DF2020
&ME	0.77	0.3	0	0.704713	0.403156	#DF2020
FORMAT:B	0.82	0.41	0	0.704713	0.403156	#DF2020
ANDHIM	0.7	0.45	0	0.704713	0.403156	#DF2020
Adam Port	0.73	0.33	0	0.704713	0.403156	#DF2020
Jesper Ryom	0.65	0.5	0	0.704713	0.403156	#DF2020
Schiller	0.58	0.34	0	0.704713	0.403156	#DF2020
Mano Le Tough	0.58	0.39	0	0.704713	0.403156	#DF2020
Anja Schneider	0.77	0.32	0	0.704713	0.403156	#DF2020
Machinedrum	0.74	0.41	0	0.704713	0.403156	#DF2020
Dennis Shepherd	0.52	0.45	0	0.704713	0.403156	#DF2020
Dirty Doering	0.64	0.45	0	0.704713	0.403156	#DF2020

Mat.Joe	0.83	0.43	0	0.704713	0.403156	#DF2020
Rampue	0.73	0.34	0	0.704713	0.403156	#DF2020
NU	0.56	0.42	0	0.704713	0.403156	#DF2020
Westbam	0.78	0.41	0	0.704713	0.403156	#DF2020
Sascha Funke	0.77	0.3	0	0.704713	0.403156	#DF2020
Lexy & K-Paul	0.78	0.46	0	0.704713	0.403156	#DF2020
Martin Roth	0.61	0.37	0	0.704713	0.403156	#DF2020
Pfaffendorf	0.75	0.42	0	0.704713	0.403156	#DF2020
D-Nox	0.72	0.33	0	0.704713	0.403156	#DF2020
Hunee	0.72	0.34	0	0.704713	0.403156	#DF2020
Steffi	0.67	0.44	0	0.704713	0.403156	#DF2020
Tiefschwarz	0.62	0.42	0	0.704713	0.403156	#DF2020
Mari Ferrari	0.71	0.34	0	0.704713	0.403156	#DF2020
Chaim	0.73	0.34	0	0.704713	0.403156	#DF2020
Marcus Meinhardt	0.77	0.38	0	0.704713	0.403156	#DF2020
Cassy	0.82	0.45	0	0.704713	0.403156	#DF2020
Frivolous	0.67	0.51	0	0.704713	0.403156	#DF2020

Aquarius Heaven	0.67	0.33	0	0.704713	0.403156	#DF2020
Donato Dozzy	0.69	0.38	0	0.704713	0.403156	#DF2020
Mathew Jonson	0.65	0.49	0	0.704713	0.403156	#DF2020
Alex Niggemann	0.78	0.48	0	0.704713	0.403156	#DF2020
Sebo K	0.85	0.45	0	0.704713	0.403156	#DF2020
Mouse on Mars	0.47	0.44	0	0.704713	0.403156	#DF2020
Re.You	0.74	0.38	0	0.704713	0.403156	#DF2020
Jan Driver	0.7785	0.37625	0	0.704713	0.403156	#DF2020

Tabelle 1: Cluster 1

Die Künstler im zweiten Cluster:

Artist	**Danceability**	**Valence**	**cluster**	**cen_x**	**cen_y**	**c**
Paul van Dyk	0.49	0.2	1	0.345455	0.142727	#81DF20
Nils Frahm	0.32	0.2	1	0.345455	0.142727	#81DF20
David August	0.2	0.09	1	0.345455	0.142727	#81DF20
Modeselektor	0.06	0.3	1	0.345455	0.142727	#81DF20
Marcel Dettmann	0.52	0.17	1	0.345455	0.142727	#81DF20
Mind Against	0.33	0.07	1	0.345455	0.142727	#81DF20
Clark	0.27	0.13	1	0.345455	0.142727	#81DF20
alva noto	0.32	0.04	1	0.345455	0.142727	#81DF20
Marusha	0.42	0.2	1	0.345455	0.142727	#81DF20
Objekt	0.44	0.11	1	0.345455	0.142727	#81DF20
Stephan Hinz	0.43	0.06	1	0.345455	0.142727	#81DF20

Tabelle 2: Cluster 2

Die Künstler im dritten Cluster:

Artist	**Danceability**	**Valence**	**cluster**	**cen_x**	**cen_y**	**c**
Claptone	0.7	0.59	2	0.777778	0.617778	#2095DF
Wankelmut	0.74	0.58	2	0.777778	0.617778	#2095DF
Robosonic	0.82	0.67	2	0.777778	0.617778	#2095DF
Henrik Schwarz	0.74	0.61	2	0.777778	0.617778	#2095DF
Session Victim	0.69	0.6	2	0.777778	0.617778	#2095DF
Sidney Charles	0.8	0.53	2	0.777778	0.617778	#2095DF
Catz n Dogz	0.82	0.6	2	0.777778	0.617778	#2095DF
FEATHERED SUN	0.88	0.86	2	0.777778	0.617778	#2095DF
Daniel Steinberg	0.83	0.55	2	0.777778	0.617778	#2095DF
Dixon	0.8	0.53	2	0.777778	0.617778	#2095DF
Jazzanova	0.57	0.61	2	0.777778	0.617778	#2095DF
Alexander Marcus	0.74	0.63	2	0.777778	0.617778	#2095DF
Ian Pooley	0.8	0.72	2	0.777778	0.617778	#2095DF
Santée	0.81	0.79	2	0.777778	0.617778	#2095DF
David Keno	0.81	0.5	2	0.777778	0.617778	#2095DF

Isolee	0.95	0.64	2	0.777778	0.617778	#2095DF
DJ T.	0.76	0.54	2	0.777778	0.617778	#2095DF
Jean Elan	0.74	0.57	2	0.777778	0.617778	#2095DF

Tabelle 3: Cluster 3

Die Künstler im vierten Cluster:

Artist	**Danceability**	**Valence**	**cluster**	**cen_x**	**cen_y**	**c**
Tale of Us	0.69	0.14	3	0.709286	0.185357	y
Oliver Koletzki	0.74	0.15	3	0.709286	0.185357	y
Tube & Berger	0.75	0.25	3	0.709286	0.185357	y
Pan-Pot	0.74	0.17	3	0.709286	0.185357	y
Apparat	0.57	0.25	3	0.709286	0.185357	y
Fritz Kalkbrenner	0.65	0.27	3	0.709286	0.185357	y
Booka Shade	0.72	0.22	3	0.709286	0.185357	y
AKA AKA	0.73	0.03	3	0.709286	0.185357	y
Rampa	0.67	0.17	3	0.709286	0.185357	y
Recondite	0.75	0.23	3	0.709286	0.185357	y
Sascha Braemer	0.72	0.21	3	0.709286	0.185357	y
Ellen Allien	0.7	0.16	3	0.709286	0.185357	y
Einmusik	0.69	0.14	3	0.709286	0.185357	y
Thomas Schumacher	0.69	0.26	3	0.709286	0.185357	y
Monika Kruse	0.72	0.12	3	0.709286	0.185357	y

Pupkulies & Rebecca	0.8	0.26	3	0.709286	0.185357	y
M.A.N.D.Y	0.79	0.16	3	0.709286	0.185357	y
Niko Schwind	0.72	0.16	3	0.709286	0.185357	y
efdemin	0.63	0.15	3	0.709286	0.185357	y
Andreas Henneberg	0.84	0.23	3	0.709286	0.185357	y
Alfred Heinrichs	0.79	0.22	3	0.709286	0.185357	y
Marc Houle	0.8	0.27	3	0.709286	0.185357	y
Current Value	0.57	0.15	3	0.709286	0.185357	y
Protonica	0.55	0.08	3	0.709286	0.185357	y
Jonas Saalbach	0.69	0.13	3	0.709286	0.185357	y
Namito	0.74	0.15	3	0.709286	0.185357	y
Scuba	0.75	0.27	3	0.709286	0.185357	y
Simina Grigoriu	0.66	0.19	3	0.709286	0.185357	y

Tabelle 4: Cluster 4

5.2.3 Analyse

Es sind vier Cluster gegeben. Dabei vereint Cluster Eins mit insgesamt 40 DJs die größte Menge an Daten in sich. Obwohl die Datenmenge vergleichsweise groß ist, ist die Streuung vom Zentrum - gemeinsam mit Cluster Vier - die geringste. Interpretieren könnte man das wie folgt: DJs in diesem Cluster haben womöglich größere Ähnlichkeiten in ihrer Musik als DJs innerhalb anderer Cluster. Die Danceability und die Valence sind hierbei hoch, jedoch insgesamt geringer als bei Cluster Drei.

Cluster Zwei beinhaltet mit lediglich elf DJs die geringste Anzahl an Daten. Die Danceability und die Valence haben hierbei die niedrigsten Werte im Vergleich zu den drei restlichen Clustern. Es ist zu erkennen, dass die Streuung vom Zentrum aus im Vergleich zu den anderen Clustern die größte ist. Die Werte gehen dabei teilweise stark auseinander. Darunter befindet sich auch der DJ mit dem niedrigsten Wert hinsichtlich der Valence.

Cluster Drei besitzt die zweitniedrigste Menge an Daten - es sind 18 DJs in diesem Cluster inbegriffen. Das Feature Danceability ist auf ungefähr demselben Stand wie Cluster Eins, die Valence besitzt jedoch den höchsten Wert der Cluster. Die Streuung ist höher als von Cluster Eins und Vier, jedoch geringer als bei Cluster Zwei. Es sind zwei DJs zu erkennen welche den höchsten Wert für die Danceability sowie die Valence besitzen. Das Zentrum des Clusters besitzt ebenfalls die höchsten Werte für die Danceability und Valence. Die Hörerschaft könnte denen aus Cluster Eins und Zwei ähneln.

Cluster Vier besitzt die zweitgrößte Datenmenge mit Daten von 28 DJs. Die Streuung ist mit Cluster Eins die geringste, die Danceability hat den zweithöchsten Wert, die Valence den zweitniedrigsten. Die Hörerschaft könnte denen von Cluster Zwei und Drei ähneln.

Es ist insgesamt zu erkennen, dass die ausgewählten DJs größtenteils Musik mit einem hohen Wert des Features Danceability produzieren. Nur ein kleiner Teil von 100 Künstlern hat dabei niedrige Werte. Andererseits ist zu erkennen, dass ein größerer Teil Musik mit niedriger Valence produziert.

Es lässt vermuten, dass Danceability ein wichtiges Feature für die Musikrichtung Techno ist und Valence zwar auch eine Rolle spielt, jedoch eine etwas geringere. Die Zentren von Cluster Eins, Drei und Vier befinden sich in dem Bereich von 0,6-0,8 bei Danceability. Hingegen befindet sich nur Cluster Drei bei Valence im Bereich von 0,6-0,8. Cluster Eins, Zwei und Vier sind unter 0,5.

Ein gesamter Vergleich zur Veränderung der Musik in der Zeit bevor COVID-19/nach COVID-19 stellt sich als schwierig dar. Aufgrund der Analysemethode sind keine einzelne Daten vorhanden, die Daten für den gesamten Zeitraum wurden zusammengelegt. Folglich ist ein Vergleich beim Clustering nicht möglich.

Die teilweise hohe Streuung innerhalb der Cluster könnte Auswirkungen auf die Regressionsanalyse haben.

6 Regression

6.1 Methodologie

Ausgehend von der getroffenen Annahme, dass durch die Schließung von Clubs und Festivals die Tanzbarkeit und durch Corona im Allgemeinen die positive Stimmung von Songs abnimmt, soll geprüft werden, ob diesbezüglich ein Zusammenhang vorliegt. Dafür wird eine Regressionsanalyse durchgeführt, bei der sich eine Trendlinie abzeichnen sollte. Bei diesem Analyseverfahren „verwendet man eine unabhängige Variable, auch Prädikator genannt, um eine abhängige Variable oder Zielvariable vorherzusagen“ [8]. Neben der Prognoseerstellung ermöglicht Regression jedoch ebenfalls, Zusammenhänge zwischen Attributen zu beschreiben oder auch einzelne Werte nur zu schätzen, wobei in diesem Fall die Deskription im Vordergrund steht.
Es gibt zwei Arten dieses Analyseverfahrens: Einerseits die einfache Regressionsanalyse, bei welcher von einer unabhängigen Variable auf eine abhängige abgebildet wird, als auch die multiple Regressionsanalyse, bei der ein Zusammenhang zwischen mehreren Prädikatoren und einer abhängigen Variable geprüft wird. Die abhängige Variable besteht immer aus einem stetigen Wert, die unabhängige kann sowohl ein stetiger als auch ein kategorialer Wert sein.
Welches Attribut abhängig und welches unabhängig ist, muss vorher geklärt werden. Beispielsweise gibt es einen Zusammenhang zwischen dem Wetter und dem Absatz einer Eisdiele, wobei das Wetter die unabhängige Eigenschaft ist, welche den Absatz beeinflusst und nicht umgekehrt [9].
Letztendlich ist das Ziel des Verfahrens, eine Beziehung der Variablen durch eine Gerade zu beschreiben, wobei der Abstand zu den gemessenen Werten möglichst gering sein soll. Die Form der Geradengleichung für die einfache Regression lautet dabei:

$$f(x) = a*x + b$$

f(x) ist hierbei der Regressand, x der Regressor und a steht für den Regressionskoeffizienten. Bei der multiplen Regressionsanalyse würden noch weitere unabhängige Variablen mit entsprechenden Koeffizienten vorhanden sein.
Ein wichtiger Wert in der Regressionsanalyse ist der Korrelationskoeffizient r, welcher einen Wert von -1 bis 1 annehmen kann. Ein Wert nahe den Intervallgrenzen deutet auf eine starke Korrelation hin und damit einhergehend auf eine geringere Streuung der Messwerte. Ermittelt wird dieser Koeffizient über die Methode der kleinsten Quadrate.
Im Folgenden sind ein paar Beispiele für die Verteilung von Messwerten anhand verschiedener r-Werte abgebildet:

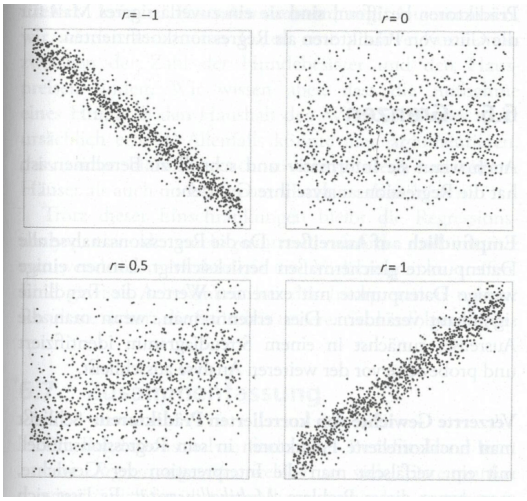

Abbildung 6: Visualisierung verschiedener r-Wert [10]

Jedoch gibt es auch Grenzen der Regressionsanalyse. Zum einen können extreme Ausreißer die Trendlinie signifikant verändern, da alle Datenpunkte gleichermaßen berücksichtigt werden. Zum anderen liefert sie keine Kausalzusammenhänge. Es können zufälligerweise Korrelationen zwischen verschiedenen Variablen entdeckt werden, wo allerdings gar keine Kausalität besteht. Somit muss man die Ergebnisse dieses Analyseverfahrens stets auf die Plausibilität prüfen.
Zudem besteht die Möglichkeit, dass eine unabhängige Variable (x) von einer bisher nicht entdeckten Variable (z) beeinflusst wird, wodurch die abhängige Variable (y) eigentlich nicht primär durch x, sondern z bedingt wird (z → x → y). Weitere Verwechslungsmöglichkeiten oder Fehler bei der Bestimmung von abhängiger und unabhängiger Variable können bei folgenden Konstellationen auftreten.

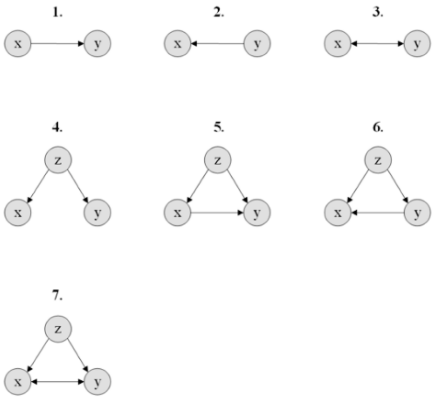

Abbildung 7: Zusammenhang abhängige und unabhängige Variable [11]

Ein erkennbarer Zusammenhang ist also zunächst lediglich als Hypothese für einen Kausalzusammenhang anzusehen.

6.2 Durchführung

6.2.1 Datenbereinigung

Trotz einiger Schwächen stellt die Regressionsanalyse eine der häufigsten Analysemethoden dar, da sie einfach anzuwenden ist und vor allem erste Hinweise auf Zusammenhänge ermittelt. Deswegen wurde sich für diese Methode entschieden und die praktische Umsetzung mithilfe von Python vorgenommen. Unsere Annahme hierbei ist, dass der Verlauf der Coronapandemie und die Lockdowns die primären Gründe für die Veränderung der von uns gewählten Features sind. Andere potentiellen Gründe für eine geringere „Tanzbarkeit“ oder „Positivität“ könnten bei einigen Künstlern der Wechsel des musikalischen Stils aufgrund persönlichen Interesses und unabhängig von COVID-19 sein, dies ist allerdings bei der Menge an Artists (n=100) zu vernachlässigen. Wie eingangs erwähnt, steht bei der vorliegenden Regressionsanalyse eher die Beschreibung der Veränderung der beiden Zielvariablen im Vordergrund und nicht eine Gleichung, welche weitere Werte prognostizieren soll, vor allem da man von einer großen Streuung der Messwerte ausgehen muss.
Die in den vorherigen Schritten erstellten Arrays müssen zuerst bereinigt werden, da diese noch viele NULL-Werte enthalten - also Tage, an denen keine neuen Songs veröffentlicht worden sind. Dafür verwendet wird dieser einfache Java Code, der als Ergebnis die neuen Arrays erstellt.

```
public class DataScienceFinal {

	public static void main(String[] args) {
		double[] valence = {};
		double[] danceability = {};
		double[] v2 = new double[500];
		double[] d2 = new double[500];
		String[] tage = {};
		String[] t2 = new String[500];
		int j = 0;
		int k = 0;

		System.out.println("valence: ");
		// Löschen der NULL-Werte im valence Array
		for (int i = 0; i < valence.length; i++) {
			if (valence[i] != 0.0) {
				v2[j] = valence[i];
				t2[j] = tage[i];
				System.out.print(v2[j] + ", ");
				j++;
			}
		}
		System.out.println();

		System.out.println("danceability: ");
		// Löschen der NULL-Werte im danceability Array
		for (int i = 0; i < danceability.length; i++) {
			if (danceability[i] != 0.0) {
				d2[k] = danceability[i];
				System.out.print(d2[k] + ", ");
				k++;
			}

		}

		System.out.println();

		System.out.println("Datum: ");
		// Löschen der Tage, wo keine Daten vorliegen
		for (int i = 0; i < t2.length; i++) {
			System.out.print("\"" + t2[i] + "\",");
		}

		System.out.println();
		System.out.println("Anzahl Einträge im Array = " + k);
	}
}
```

Abbildung 8: Java Code zum eliminieren von Null-Einträgen

6.2.2 Modellerstellung

Im Folgenden kann man mit der Modellerstellung beginnen, in diesem Fall ist dies eine einfache Regressionsanalyse, einmal für die Tanzbarkeit und einmal für die Positivität der Lieder.

Um einen Überblick zu gewinnen, wurde mit Python ein Streudiagramm für beide Features erstellt, um die Verteilung der einzelnen Messwerte zu visualisieren. Die x-Achse steht für den von uns untersuchten Zeitraum vom 1.1.2018 bis zum 23.06.2021, allerdings sind ausschließlich die Tage eingetragen, an denen Messdaten vorliegen. Die y-Achse steht bei Abbildung 4 für die Danceability-Werte im Intervall von 0 bis 1 und bei Abbildung 5 für die Valence-Werte, auch im Intervall 0 bis 1.

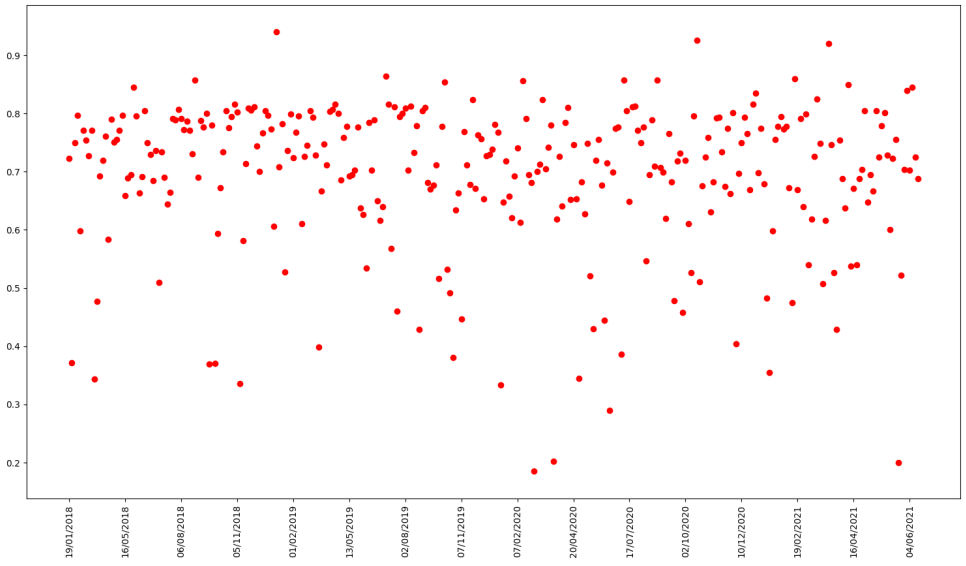

Abbildung 9: Danceability

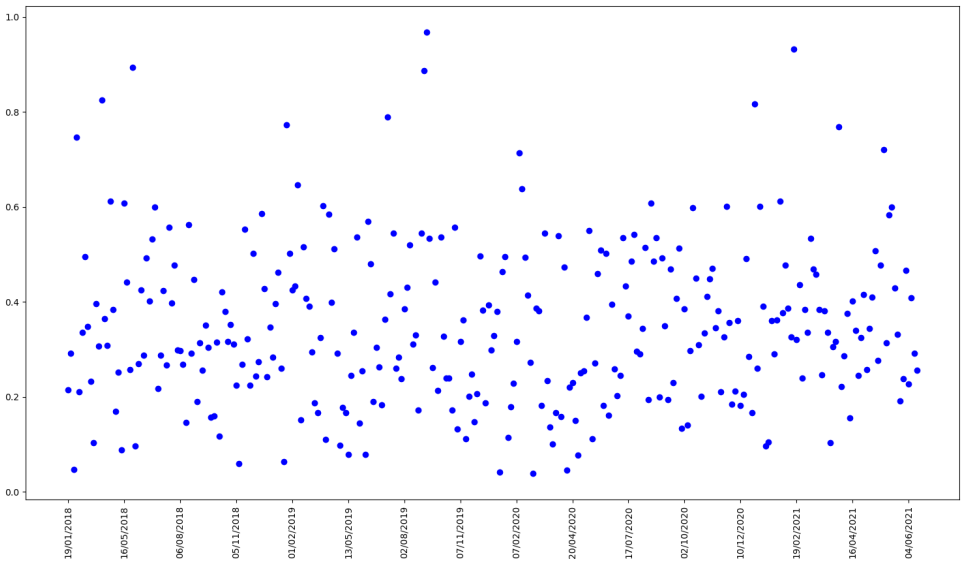

Abbildung 10: Valence

Wie in beiden Grafiken zu erkennen ist, liegt eine sehr hohe Streuung der Werte vor. Um jetzt einen potentiellen Trend erkennen zu können, wird nun eine Regressionsanalyse durchgeführt, indem der Python Code erweitert wird (siehe Abb. 6). Die Arrays in der Abbildung sind aus Platzgründen leer dargestellt.

```
from scipy.stats import linregress
import matplotlib.pyplot as plt

danceability = []
valence = []
datum = []
tage = []
b, a, r, p, std = linregress(tage,danceability)
b2, a2, r2, p2, std2 = linregress(tage,valence)
print("danceability/Datum -> ","Steigung: " , b, "Achsenabschnitt: ",a,"Korrelationskoeffizient
(Pearson): ",r,"p-Wert :" ,p,"Standardabweichung ",std)
print("valence/Datum -> ", "Steigung: " , b2, "Achsenabschnitt: ",a2,"Korrelationskoeffizient
(Pearson): ",r2,"p-Wert :" ,p2,"Standardabweichung ",std2)
plt.scatter(datum,danceability)
plt.scatter(datum,valence)

plt.title('Regressionsanalyse')
plt.plot([0,len(tage)],[a,a+len(tage)*b],c="red",alpha=0.9)
plt.plot([0,len(tage)],[a2,a2+len(tage)*b2],c="blue",alpha=0.9)
plt.plot()
plt.xlim(0,len(tage))
plt.ylim(0,1.0)
plt.xlabel("Datum/Verlauf")
plt.ylabel("avg danceability(rote Gerade)/avg valence(blaue Gerade)")
plt.grid(alpha=0.9)
plt.xticks([x for x in range(len(datum)+1) if x%20==0])
plt.xticks(rotation=90)
plt.show()
```

Abbildung 11: Verwendeter Python Code

Als Ergebnis werden folgende Werte und folgendes Diagramm geliefert:

Regressionsgerade danceability/Datum →

Steigung: -0.0001163112960669503

Achsenabschnitt: 0.7123324845604522

Korrelationskoeffizient (Pearson): -0.07954871014983833

p-Wert : 0.16652429889591744

Standardabweichung: 8.3870031913759e-05

Regressionsgerade valence/Datum →

Steigung: 6.494809760997635e-05

Achsenabschnitt: 0.3396623678644549

Korrelationskoeffizient (Pearson): 0.03365224811525528

p-Wert : 0.5588870679341935

Standardabweichung: 0.00011099484984985255114

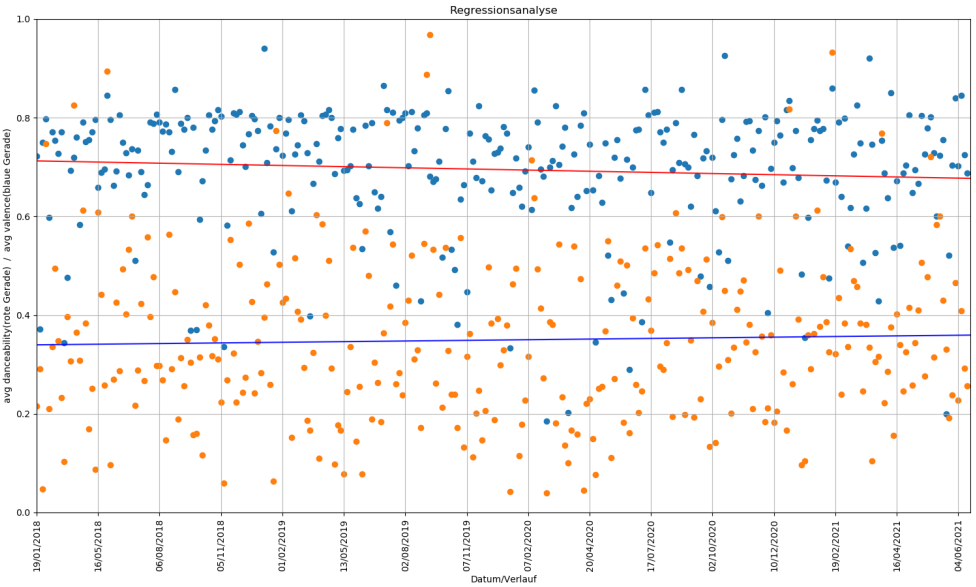

Abbildung 12: Regressionsanalyse Python Plot

Des Weiteren wird eine Analyse für den Zeitraum bis zum ersten Lockdown in Deutschland (23.03.2020) und ab diesem erstellt.

Vor dem Lockdown:

danceability/Datum →

Steigung: -0.00018373818181050257

Achsenabschnitt: 0.7201752406734826

Korrelationskoeffizient (Pearson): -0.07507918121166485

p-Wert : 0.32765082633205744

Standardabweichung 0.00018716641827088709

valence/Datum →

Steigung: -8.346678647647118e-05

Achsenabschnitt: 0.354719790337049

Korrelationskoeffizient (Pearson): -0.023290173139300818

p-Wert : 0.7616904933039381

Standardabweichung: 0.00027478859012563194

Ab dem Lockdown:

danceability/Datum →

Steigung: 0.00030710305626060997

Achsenabschnitt: 0.6615503656514824

Korrelationskoeffizient (Pearson): 0.0862740422204627

p-Wert : 0.32530185915692666

Standardabweichung: 0.00031103552721325144

valence/Datum →

Steigung: 0.000696082738924671

Achsenabschnitt: 0.30597086564345866

Korrelationskoeffizient (Pearson): 0.16844406994913158

p-Wert : 0.05352175884416482

Standardabweichung 0.0003572589840928644

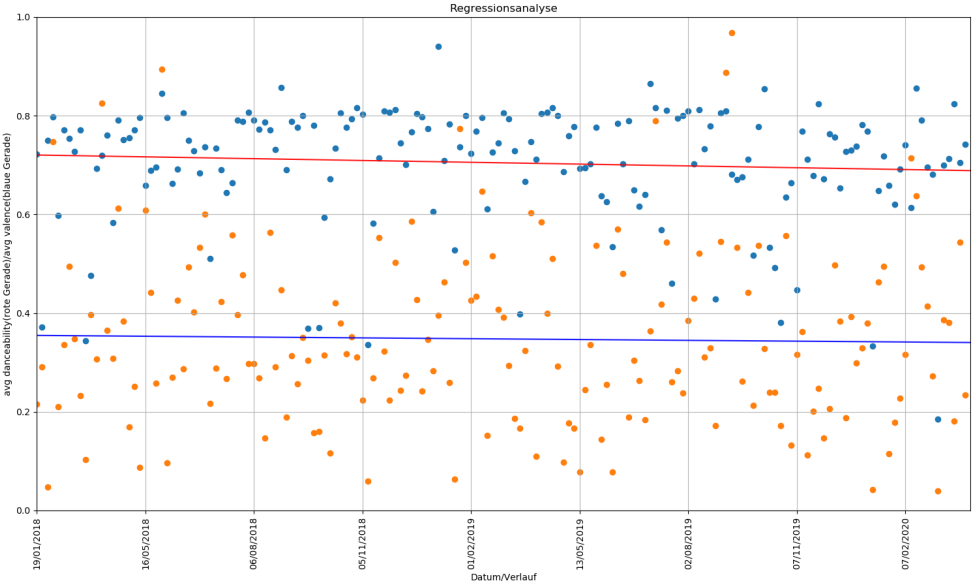

Abbildung 13: Python Plot bis 1. Lockdown

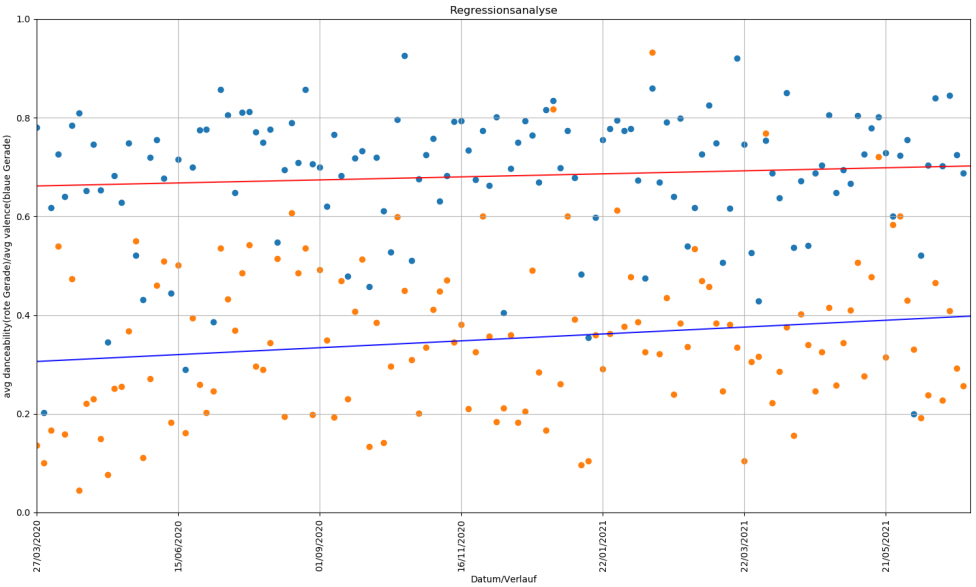

Abbildung 14: Python Plot ab 1. Lockdown

6.2.3 Analyse / Ergebnisse

Entgegen der getroffenen Annahme gibt es über den gesamten Zeitraum keine signifikante Änderung der Features. Der Abwärtstrend bei der Tanzbarkeit ist minimal und hat nur einen sehr kleinen, negativen Korrelationskoeffizienten (r=-0.079), während die Positivität in den Liedern sogar leicht zunimmt, allerdings auch nur sehr schwach. Somit kann, zumindest anhand der gewählten Methodik, keine wirkliche Korrelation zwischen der Tanzbarkeit beziehungsweise Positivität und dem zeitlichen Verlauf gezeigt werden.
Deswegen haben wir den Untersuchungszeitraum nochmals aufgeteilt, da eventuell ein stärkerer Zusammenhang erst mit dem Beginn des ersten Lockdowns auftreten könnte. Jedoch ist zu beobachten, dass die Gerade beider Features ab dem Lockdown sogar ansteigt. Auch hier ist bei der Valence der Korrelationskoeffizient nicht wirklich stark ausgeprägt (r=0,168), allerdings ist dieser leichte Trend schon unerwartet. Vor dem Lockdown besitzt die Gerade sogar eine sehr kleine negative Steigung.
Ein ähnliches Bild zeichnet sich bei der Danceability ab. Auch hier gibt es einen Abwärtstrend vor dem Lockdown und einen Anstieg danach - allerdings ist beides so minimal, dass sich dies nicht wirklich für eine treffende Aussage eignet, zudem liegt in beiden Fällen auch der r-Wert nahe Null.
Insgesamt kann man festhalten, dass es keinen wirklichen Zusammenhang zwischen dem Verlauf von COVID-19 und den gewählten Features in elektronischer Musik gibt, zumindest wenn man 100 Berliner DJs als Referenzmenge nimmt. Die Veränderungen über die Zeit hinweg sind so klein, dass diese wohl eher dem Zufall geschuldet sind. Deswegen ist die untersuchte Fragestellung mit einem klaren “Nein” zu beantworten.

7 Zusammenfassung

Die Ergebnisse die wir unter der Forschungsfrage “Wie wirkt sich COVID-19 und die damit einhergehende Schließung von Clubs/Festivals auf die Tanzbarkeit und die Positivität von elektronischer Musik aus?” erwartet haben sind nicht eingetreten. Deshalb kann die generelle These, dass die Gesellschaft Musik beeinflusst und anders herum, nicht bestätigt werden. Dies kann an einer falschen Auswahl der Parameter für die vorliegende Forschungsfrage liegen, es ist aber auch möglich, dass sich Veränderungen erst nach einem bedeutend längeren Zeitraum zeigen. Dadurch ist die Empfehlung ein länger anhaltendes Ereignis (10 Jahre aufwärts) mit ähnlichen Features zu untersuchen, um echte Trends erkennen zu können.
Bei der gesamten Untersuchung ist sich an den Semestermateialien von Prof. Dr. Christin Schmidt orientiert worden. Somit konnte das 8-Phasen-Modell einer Datenanalyse verwendet werden, um valide Ergebnisse liefern zu können. Hinsichtlich der beiden Spotify Features Danceability und Valence liegen keine ausreichenden Veränderungen vor, um diese in Zusammenhang mit COVID-19 setzen zu können.

8 Literaturverzeichnis/Quellen

[1] ”Zitate berühmter Personen”, https://beruhmte-zitate.de/zitate/131609-martin-scherber-die-musik-soll-alles-befruchten-in-ihr-sind-die-w/, 22.06.2021.

[2] Doris Bammer: “Die Wirkung von Musik auf den Menschen”, https://www.grin.com/document/103180 file:///Users/maxhager/Downloads/MusikinGesellschaftund Politik Brunner 2010.pdf., 22.06.2021.

[3] Sophia Seidel, “Musik und Emotionen”, https://monami.hs-mittweida.de/frontdoor/deliver/index/docId/6471/file/BACHELORARBEIT+Sophia+Seidel.pdf, 22.06.2021.

[4] Doris Bammer: “Die Wirkung von Musik auf den Menschen”, https://www.grin.com/document/103180, 23.06.2021.

[5] https://thedjlist.com/world/germany/berlin/djs/, 24.06.2021.

[6] “Web API Reference”, https://developer.spotify.com/documentation/web-api/reference/, 24.06.2021.

[7] Paul Lamere: „Welcome to Spotipy!”, https://spotipy.readthedocs.io/en/2.18.0/, 24.06.2021.

[8] Annalyn Ng, Kenneth Soo: Data Science-was ist das eigentlich?! Springer, 2017, S.73.

[9] Christin Schmidt: Grundlagen Data Science, Foliensatz Statistik III.

[10] Annalyn Ng, Kenneth Soo: Data Science-was ist das eigentlich?! Springer, 2017, S.83.

[11] http://www.methoden-psychologie.de/elearning/ursachen_korrelation.png, 24.06.2021.

[12] Clusteranalyse, https://de.wikipedia.org/wiki/Clusteranalyse, 28.06.2021.

[13] Machine Learning II (Unsupervised Learning) Seminararbeit als modulbegleitende Prüfungsleistung für das Modul: Grundlagen sozialer Netze an der Hochschule für Technik und Wirtschaft (HTW) Berlin Fachbereich 4 Informatik, Kommunikation und Wirtschaft Studiengang Angewandte Informatik, S.21 https://moodle.htw-berlin.de/pluginfile.php/1122457/mod_resource/content/1/10%20Machine%20Learning%20II_Unsupervised_ML.pdf, 28.06.2021.

[14] K-Means-Algorithmus, https://de.wikipedia.org/wiki/K-Means-Algorithmus, 28.06.2021.

[15] Machine Learning II (Unsupervised Learning) Seminararbeit als modulbegleitende Prüfungsleistung für das Modul: Grundlagen sozialer Netze an der Hochschule für Technik und Wirtschaft (HTW) Berlin Fachbereich 4 Informatik, Kommunikation und Wirtschaft

Studiengang Angewandte Informatik, S.21
https://moodle.htw-berlin.de/pluginfile.php/1122457/mod_resource/content/1/10%20Machine%20Learning%20II_Unsupervised_ML.pdf, 28.06.2021.

[16] Machine Learning Clustering in Python,
https://rocketloop.de/de/blog/machine-learning-clustering-in-python/, 28.06.2021.

[17] Spotify Web API, https://developer.spotify.com/console/, 28.06.2021.

[18] Benz Andreas, Radke Eugen, „Algorithmen zur Berechnung konvexer Hüllen von Punkten“, S.3-4,
http://www.gm.fh-koeln.de/~hk/lehre/ala/ws0506/Praktikum/Projekt/C_lila/ConvexHull.pdf, 23.06.2021.

weitere Quellen:

Christin Schmidt: Grundlagen Data Science, Foliensatz Statistik III.

"Pearson Korrelation", https://studyflix.de/statistik/pearson-korrelation-1051, 19.06.2021.

"Lineare Regression in Python",
https://ichi.pro/de/lineare-regression-in-python-181212097752087, 17.06.2021.

"Mathplotbib", https://www.grund-wissen.de/informatik/python/scipy/matplotlib.html, 20.06.2021.

"Lineare Regression und Anwendung in Python",
https://statisquo.de/2018/03/23/lineare-regression-und-implementierung-in-python/, 12.06.2021.