Analyse und Interpretation der Kehrichtmengen im Kanton Zürich 1996-2021: Ein Clustering- und Two-Way Fixed Effects Modell mit Panel-Daten

Im Rahmen des Moduls "Data Mining"

an der Digital Business University of Applied Sciences DBU Berlin

Autor:	Simon Sahli
Studiengang:	Data Science und Management (M.Sc.)
Immatrikulation:	190252
Adresse:	CH-3065 Bern
E-Mail:	simon.sahli@student.dbuas.de
Datum der Einreichung:	10.09.2023

Inhaltsverzeichnis

1 Einleitung

Laut Daten des Bundesamtes für Umwelt (BAFU)[10] erzeugt die Schweizer Bevölkerung rund 700 kg Abfall pro Kopf (Deutschland: 618 kg), was zu den höchsten Mengen in Europa zählt. Der Inhalt des Hausmülls (in der Schweiz "Kehricht"genannt) spiegelt das Konsumverhalten und somit auch den wirtschaftlichen Zustand der Gesellschaft wider. Trotz Bemühungen zur Sensibilisierung der Öffentlichkeit und Verbesserungen bei den Recycling-Sammelstellen, zeigen Analysen des BAFU aus dem Jahr 2012, dass immer noch bedeutende Mengen wiederverwendbarer Materialien – insbesondere biogene Abfälle und Lebensmittel – im Müll enden. Daher ist die vom BAFU alle zehn Jahre durchgeführte Analyse der Kehrichtzusammensetzung von entscheidender Bedeutung, um zu ermitteln, wie sorgfältig und nachhaltig wir im Alltag mit Ressourcen umgehen. Die Analyse aus dem Jahr 2012 offenbart zwar gewisse Erfolge bei der Begrenzung der Müllmenge, allerdings ist gleichzeitig der Anteil der biogenen Abfälle im Haushaltsmüll von 27 % auf 32 % gestiegen. Ein grosser Teil davon besteht noch aus überwiegend essbaren Lebensmitteln. Das BAFU hat in seinem 2012er Bericht betont, dass erhebliche Unterschiede zwischen den einzelnen untersuchten Gemeinden bestehen und dass die Abfallentsorgung in den Gemeinden weiter optimiert werden muss.
Im November 2022 kündigte das BAFU bei einem Medienereignis in Perlen im Kanton Luzern an, dass die neueste Analyse der Kehrichtzusammensetzung begonnen hat. "*Die Reduzierung der Abfallmenge durch Vermeidung von Lebensmittelabfällen und Förderung von Recycling ist eine fesselnde und dringliche Herausforderung. Würde die gesamte Welt im selben Ausmass wie die Schweiz konsumieren und Abfall erzeugen, bräuchten wir fast drei Planeten. Die Kreislaufwirtschaft ist ein Teil der Lösung, um uns aus dieser Sackgasse zu führen*", sagte BAFU-Direktorin Katrin Schneeberger[3]. Die Ergebnisse der 2022er Analyse werden im Laufe der zweiten Hälfte des Jahres 2023 veröffentlicht.

Die vorliegende Studienarbeit, die im Rahmen des Moduls "Data Mining" an der Digital Business University of Applied Sciences Berlin erstellt wird, zielt darauf ab, mit Hilfe von Open Government Data des Kantons Zürich, bestehend aus Daten von 177 Gemeinden über 26 Jahre, Muster in Bezug auf die Kehrichtmenge zu identifizieren.
Es werden zwei Hypothesen aufgestellt, die in der folgenden Studienarbeit untersucht werden:

- H1: Es existieren verschiedene Gemeindegruppen, die sich hinsichtlich ihres Abfallaufkommens der Haushalte, sowie weiteren sozioökonomischen Faktoren differenzieren.
- H2: Das Abfallaufkommen der Haushalte in einer Gemeinde wird durch Faktoren wie Arbeitslosigkeit, Ausgaben für den Umweltschutz pro Kopf, Bildungsgrad und die politischen Präferenzen beeinflusst.

Für die Erarbeitung dieser Hypothesen wird der "Cross Industry Standard Process for Data Mining (CRISP-DM)" verwendet, wie er in der Arbeit von Shearer (2000)[11] vorgestellt wurde.
Der erste Teil dieser Arbeit stellt das Geschäftsziel und die Projektplanung kurz dar, um daraus die Probleme abzuleiten, die mit Hilfe der Datenbasis analysiert werden sollen. Im zweiten Teil wird diese Datenbasis im Rahmen einer explorativen Datenanalyse genauer untersucht und vorgestellt. Im nächsten Kapitel werden Methoden der Clusteranalyse und Regression vorgestellt und modelliert, mit denen die erwähnten Hypothesen untersucht werden sollen.
Eine interaktive Übersicht über dieses Data-Mining Projekt, ein Screencast so wie auch die Datengrundlage ist unter folgendem Link einsehbar:
`https://www.simonsahli.ch/project_kehricht.html`

2 Problemstellung

Das BAFU widmet sich intensiv der Aufgabe, den Zustand und die Verhaltensweisen in Bezug auf die Abfallentsorgung zu analysieren und einzuschätzen. Dies beinhaltet eine detaillierte Untersuchung ausgewählter Haushaltsabfälle aus verschiedenen Gemeinden, aus welchen Muster abgeleitet und auf die gesamte Schweiz hochgerechnet werden. Obwohl die Studien des BAFU äusserst präzise und relevant sind, beziehen sie sich zwangsläufig nur auf eine begrenzte Anzahl von Proben.

Um diesen Ansatz des BAFU zu ergänzen, kommen in der vorliegenden Studie Methoden der Knowledge Discovery in Databases (KDD) und des Data Mining zur Anwendung. Fayyad, Piatetsky-Shapiro und Smyth (1996)[6] heben in ihrer Arbeit die Bedeutung von KDD hervor, das dazu dient, High-Level-Wissen aus umfangreichen Low-Level-Datensätzen zu extrahieren. Data Mining, so erklären die Autoren, ist dabei eine Anwendung spezifischer Algorithmen zur Mustererkennung in Daten.

Die Kombination des BAFU-Ansatzes und des Data Mining erlaubt ein detaillierteres Bild der Abfallsituation in einer bestimmten Region. So können spezifische Eigenschaften und Unterschiede auf Gemeindeebene identifiziert und aufgezeigt werden, dass die Abfallproduktion sowohl das Ergebnis individuellen Konsumverhaltens als auch breiter sozialer und wirtschaftlicher Bedingungen ist. Durch die Einbeziehung zusätzlicher Datensätze, etwa Durchschnittsalter der Bevölkerung, Arbeitslosenquote, politische Wähleranteile und Steuerkraft der Gemeinde in die Data Mining-Analysen, kann ein tieferes Verständnis dieser Zusammenhänge erzielt werden.

Aufgrund der variierenden Qualität und Verfügbarkeit von Daten in den verschiedenen Kantonen, konzentriert sich diese Studienarbeit auf die Region des Kantons Zürich, der eine breite Palette zuverlässiger Datenquellen bietet. Das Ziel der Arbeit ist es, ein Modell zu erstellen, das die

Menge des Haushaltabfalls in verschiedenen Gemeinden des Kantons Zürich auf der Grundlage von demographischen und sozioökonomischen Daten in Bezug setzt. Dabei kommen Regressions- und Clusteringanalysen zum Einsatz, um bisher nicht erkennbare Muster und Zusammenhänge aufzudecken. Diese Vorhersagen und Erkenntnisse können dem Kanton Zürich und dem BAFU dienen, ihre Abfallentsorgungs- und Recyclingprogramme effizienter zu gestalten. Die Ergebnisse des Modells könnten zudem dazu beitragen, das Verständnis der Faktoren, die die Abfallproduktion beeinflussen, zu vertiefen. Dies wäre nicht nur für die beteiligten Gemeinden und Behörden von Vorteil, sondern auch für die Umwelt und die Gesellschaft insgesamt.

3 Daten

Der Prozess der Datenmodellierung erfordert ein tiefgreifendes Verständnis der zugrunde liegenden Daten. In diesem Sinne konzentriert sich dieses Kapitel auf eine explorative Datenanalyse (EDA). Ziel dieser Analyse ist es, ein detailliertes Verständnis der Merkmale und statistischen Eigenschaften der beteiligten Variablen zu erlangen.

Der Datensatz, der für diese Studienarbeit verwendet wird, resultiert aus der Arbeit der Fach- und Koordinationsstelle OGD[1]. Sie ist für die Koordinierung aller offenen Behördendaten des Kantons Zürich verantwortlich und stellt diese in einem online zugänglichen Metadaten-Katalog zur Verfügung. Durch die Verwendung einer interaktiven Anwendung innerhalb des Gemeindeportals konnten die relevanten Datensätze für 177 Gemeinden des Kanton Zürichs über mehrere Jahre hinweg zusammengestellt werden. Insgesamt wurden für die folgende Datenanalyse 17 öffentlich zugängliche Datensätze auf Gemeindeebene kombiniert. Diese stammen aus verschiedenen Quellen, darunter das Statistische Amt des Kantons Zürich, das Staatssekretariat für Wirtschaft (seco), das kantonale Steueramt sowie das Amt für Abfall, Wasser, Energie und Luft (AWEL). Mit diesen Datenquellen stehen insgesamt 61'387 Datenpunkte zur Analyse bereit. Die Datenpunkte decken ein breites Spektrum ab, von Bevölkerungsdaten (Gesamtbevölkerung, Durchschnittsalter und Arbeitslosenquote) über Abfallverhalten (Kehrichtmengen, Altpapiermengen, Verpackungsglasmengen, Wasserverbrauch) bis hin zu Gemeindefinanzen und politischen Einstellungen (steuerbares Einkommen, Steuerkraft, Nettoaufwand Umweltschutz, Wähleranteile der SP und SVP). Eine vollständige Übersicht der verwendeten Datensätze je Gemeinde und Jahr ist im Anhang aufgeführt (vgl. Abbildung I im Anhang).

Um den Datensatz und seine Struktur besser zu verstehen, wurde ein sogenannter "Profiling Report" erstellt. Dieser Bericht fasst die Gesamtmenge und Struktur der Daten zusammen und dient als

[1] `https://www.zh.ch/de/politik-staat/opendata.html`

erstes Instrument zur Beurteilung der Datenqualität, indem er auf die Vollständigkeit der Daten hinweist. Im weiteren Verlauf dieses Kapitels wird eine exploratory data analysis (EDA) durchgeführt, die eine Reihe relevanter Grafiken und Analysen beinhaltet. Das Ziel dabei ist es, relevante Daten zu identifizieren, ihre Vollständigkeit zu überprüfen und die Relevanz der Datensätze für das hier behandelte Data Mining Problem zu beurteilen. Dies umfasst auch eventuell notwendige Schritte zur Datenreinigung und -transformation.

3.1 Explorative Analyse

In dem hier vorliegenden Datensatz werden 177 Gemeinden des Kanton Zürich betrachtet ("GEBIET _ NAME"). Tatsächlich verzeichnet der Kanton Zürich jedoch (Stand: Juli 2023) nur 162 politische Gemeinden. Die Ursache für diese Diskrepanz liegt in der historischen Datensammlung: Die ersten Einträge ("INDIKATOR _ JAHR") reichen bis ins Jahr 1962 zurück. Dadurch sind Gemeinden erfasst, die zwischen 1962 und 2023 fusioniert haben. Ein Beispiel hierfür ist die Gemeinde "Schönenberg". Sie war bis Ende 2018 eigenständig, gehört aber seitdem zu Wädenswil. Für die Analyse ist es wichtig zu beachten, dass einige Regionen nach einer gewissen Zeitspanne im Datensatz verschwinden, während andere durch Fusionen ab einem bestimmten Zeitpunkt signifikante Datenänderungen erfahren. Insbesondere bei der Untersuchung der Kehrichtmengen in Tonnen, ist die "Tonnen pro Kopf"-Betrachtung essenziell, damit solche Gebietsveränderungen adäquat berücksichtigt werden.

Die Erhebung der betrachteten Datensätze begann 1962. Dabei ist die Verteilung der Beobachtungen zwischen 1962 und 2023 nicht gleichmässig: Erst in den 90er Jahren nimmt die Datenmenge merklich zu. Abbildung 1 zeigt mit einem Histogramm die zeitliche Verteilung der Beobachtungen. Abbildung 2, ein Scatterplot, verknüpft den zeitlichen Kontext mit der Art des erhobenen Datensatzes. Es wird klar: Ab 1962 gibt es vorwiegend Daten zur Bevölkerungsstruktur. Erst ab 1996 wurden Gemeindedaten zum Abfall (Kehricht, Altpapier, Altglas) und ab 2006 zum Wasserverbrauch erfasst.

Abbildung 1: Verteilung aller Beobachtungen über die Zeit

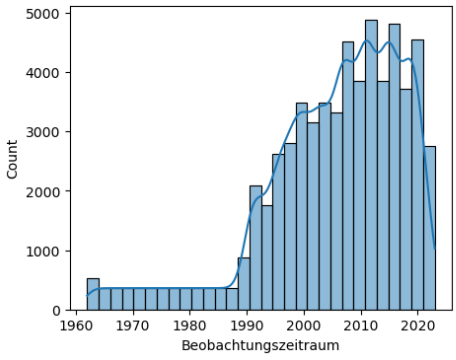
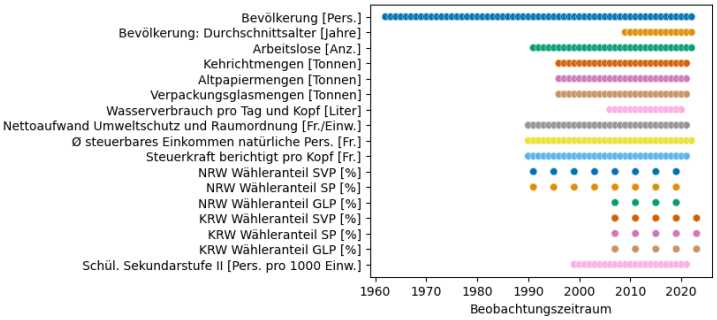


Abbildung 2: Vorhandene Daten je Datensatz über die Zeit

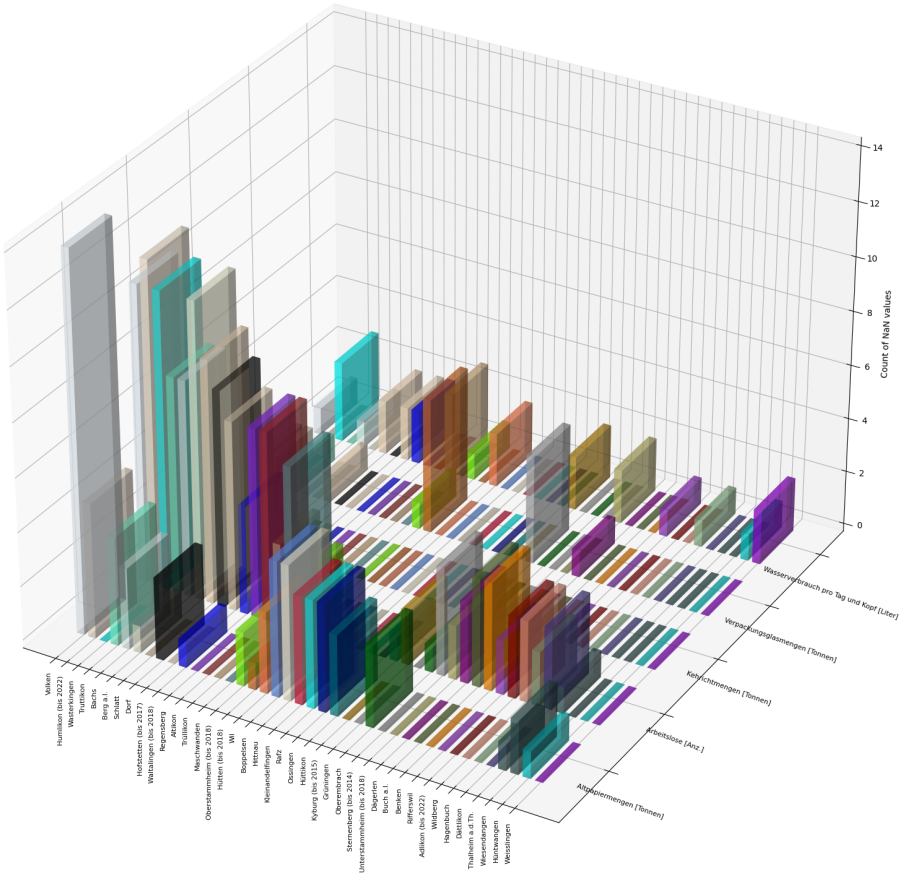


Ein wichtiger Aspekt der explorativen Analyse betrifft die Zeilen im Datensatz, die keinen Beobachtungswert ("NaN") aufweisen. Bei insgesamt 61'387 Beobachtungen treten 289 solcher Fälle auf, das entspricht weniger als 0.1 %. Trotz dieser geringen Anzahl werden diese 289 Fälle näher untersucht. Hierbei sind zwei Fragen zentral: Wo und wann treten fehlende Datenpunkte auf und welche Datensätze sind betroffen?

Abbildung A im Anhang zeigt ein Barchart, welcher die Anzahl der fehlenden Werte pro Region visualisiert. Es zeigt sich, dass in 65 Regionen Daten fehlen, wobei die meisten nur in bestimmten Jahren betroffen sind. Acht Regionen haben 10 oder mehr fehlende Werte, wobei die Region "Volken" mit 27 fehlenden Werten am stärksten betroffen ist.

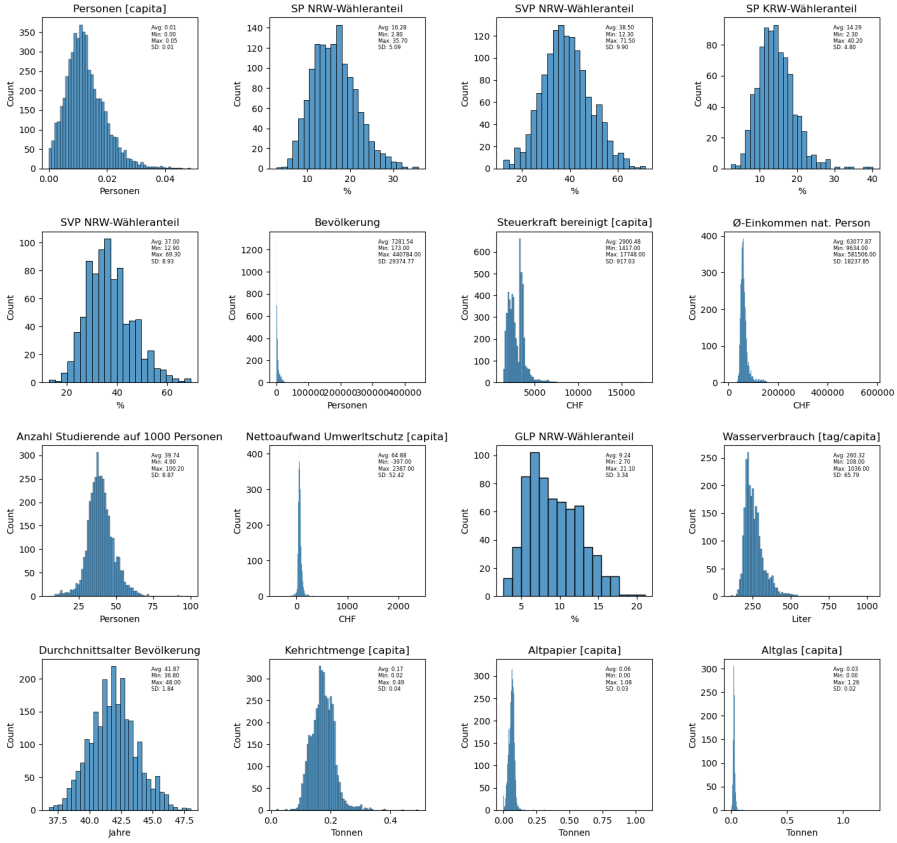
Abbildung 3 zeigt die fehlenden Werte von 40 Gemeinde und zugehörigem Datensatz mit Hilfe einer 3D-Visualisierung. Besonders auffällig ist der Mangel an Daten zu Altpapiermengen und der Anzahl Arbeitsloser, insbesondere in der Gemeinde Volken. Die Überprüfung ergibt, dass diese "NaN" Werte nicht als "Nullwerte" gelten können. Dies wird beispielsweise bei Altpapiermengen deutlich (unplausibel) und bei der Arbeitslosenzahl, da Volken 2007 eine Arbeitslosenrate von 0 % explizit auswies und sechs aufeinanderfolgende Jahre ohne Arbeitslosigkeit unwahrscheinlich sind, wie z.B. ab 2017 der Fall ist. Zusammenfassend lässt sich sagen, dass "NaN"-Zeilen im Datensatz am ehesten als "nicht verfügbar" und somit als "ungültig" gelten (s. auch Kapitel 4).

Abbildung 3: 3D-Visualisierung der fehlenden Datenwerte aus 40 Gemeinden des Kantons Zürich



Im Folgenden werden die Verteilungen und möglichen Extremwerte der einzelnen Datensätze betrachtet. Obwohl die Datenmanipulation später im Detail behandelt wird (s. Kapitel 4), wird an dieser Stelle eine Anpassung vorgenommen: Um aussagekräftige Histogramme zur Abfallsituation der Gemeinden zu erstellen, werden drei neue Merkmale eingeführt. Dabei werden die Werte Kehricht, Altpapier und Altglas in Tonnen jeweils durch die Bevölkerungszahl des betreffenden Jahres und der jeweiligen Gemeinde geteilt, um die Abfallsituation pro Kopf zu berechnen. Aggregierte Abfallvariablen je Gemeinde und Jahr werden nicht weiter berücksichtigt. Abbildung 4 gibt einen Überblick über die Verteilungen verschiedener Datensatz-Variablen.

Abbildung 4: Überblick über die Verteilungen verschiedener Datensatz-Variablen



Ausreisser werden über die 1.5*IQR-Regel ermittelt. Hierbei bezeichnen $Q1$ und $Q3$ das erste bzw. dritte Quantil der Verteilung, und IQR ist der Interquartilsabstand $(Q3 - Q1)$. Ein Wert i gilt als Ausreisser, wenn er ausserhalb des Intervalls $[Q1 - 1.5 * IQR, Q3 + 1.5 * IQR]$ liegt. Bei der Arbeitslosenvariablen zeigt sich eine hohe Standardabweichung, was auf eine starke Streuung der Datenpunkte um den Mittelwert hinweist. Insbesondere Daten aus den 90er Jahren, z.B. aus den Gemeinden “Opfikon” und “Dietikon” (1997), weisen hohe Arbeitslosenquoten pro Person auf. Nach einer kurzen Recherche im Internet bestätigt sich die Plausibilität dieser Werte, wodurch ein Datenfehler unwahrscheinlich erscheint.

Beim Blick auf den Nettoaufwand für Umweltschutz pro Person in den Gemeinden fällt ein negativer Minimalwert auf. Der Datenbeschrieb des statistischen Amtes des Kanton Zürich klärt auf: Dieser Wert ergibt sich aus der Differenz des Bruttoertrags und des Bruttoaufwands aus der Steuerrechnung. Negative Werte weisen darauf hin, dass der Ertrag höher war als der geplante Aufwand. Mit der 1.5*IQR-Regel bestätigte Ausreisser zeigen keine Auffälligkeiten im negativen Bereich. Jedoch sticht "Sternenberg" im Jahr 2014 mit CHF 2'387.00 hervor, ein Betrag, der 4.3-mal über dem nächsthöchsten Wert liegt. Angesichts der Fusion von "Sternenberg" mit "Bauma" in jenem Jahr könnte es sich um einen Buchhaltungsfehler handeln. Es wäre ratsam, diesen Wert in weiteren Analysen auszuklammern.

Weiter bei der Untersuchung von Extremwerten fallen Abweichungen im Wasserverbrauch und bei den Abfalldaten (Papier, Glas, Kehricht) auf. Besonders auffällig ist "Hüttikon" im Jahr 2006 mit 1'036 Litern Wasser pro Tag und Person, verglichen mit einem Jahresdurchschnitt von 242.19 Liter (2019) bis 280.52 Liter (2007) in anderen Gemeinden. Das statistische Amt des Kanton Zürich weist darauf hin, dass Wasserverbrauchswerte pro Kopf Industrie und Gewerbe einschliessen. Da 2006 die Fussballweltmeisterschaft in Deutschland stattfand und viele Personen in der Schweiz Public-Viewings besuchten, könnte der Wasserverbrauch gestiegen sein. Dennoch erscheint ein Wert von über 1'000 Litern pro Tag und Person extrem und ist im Datensatz einzigartig.

Bei der Betrachtung des Kehrichts steht der Fokus auf Tonnen pro Person. Der Durchschnittswert liegt über alle Jahre und Gemeinden bei 0.17 Tonnen pro Person und Jahr, was 170 kg entspricht und plausibel ist. Während einige Werte deutlich über dem Durchschnitt liegen (> 0.3 Tonnen), erscheinen insbesondere neun Beobachtungen mit Werten < 0.08 Tonnen auffällig. "Horgen" zeigt 2006 mit 0.0173 Tonnen den niedrigsten Wert. Dieser Wert scheint, auch in Anbetracht der Werte der umliegenden Jahre, unplausibel zu sein. Bei Altpapier liegt der Durchschnitt bei 60kg pro Person und Jahr und bei Altglas bei 30kg. Einzelheiten dazu sind in den entsprechenden Abschnitten im Programmcode ersichtlich.

Zum Abschluss der explorativen Analyse des Datensatzes dient die Abbildung B im Anhang, eine Heatmap der Korrelationen. Hier lassen sich die Datenverteilungen erneut überprüfen. Es zeigt sich eine starke positive Korrelation zwischen den Wähleranteilen der schweizerischen Volkspartei (SVP) für Kantons- und Nationalrat. Gleichzeitig korrelieren die Wähleranteile der SVP stark negativ mit denen der Sozialdemokratischen Partei (SP). Diese Ausschläge waren alle zu erwarten und bestätigen eine gewisse Plausibilität in der Datenerhebung.

4 Theorie und Modellierung

In diesem Kapitel steht die Anwendung zweier Analysemethoden im Mittelpunkt: Clustering (unüberwachtes Lernen) und Regression (überwachtes Lernen), die auf den zuvor vorgestellten Datensatz angewendet werden.

Clustering erweist sich als wertvolles Instrument, um nicht offensichtliche Muster und Beziehungen in den Daten zu identifizieren. Ziel ist es, die Gemeinden basierend auf ihrem Abfallverhalten sowie demografischen und sozioökonomischen Merkmalen in Cluster zu gruppieren. Dies soll Aufschluss darüber geben, ob im Kanton Zürich Gemeindegruppen mit ähnlichen Profilen existieren. Die Erkenntnisse aus dieser Clustering-Analyse bieten eine Grundlage, um Hypothesen für die anschliessende Regressionsanalyse zu entwickeln. Bei dieser Analyse wird die Beziehung zwischen einer Zielvariable und einer oder mehreren erklärenden Variablen modelliert.

Zusammengefasst ermöglicht die Kombination von Clustering und Regressionsanalyse ein tiefgehendes Verständnis für Muster in den Daten. Dies kann helfen, massgeschneiderte Strategien zur Abfallreduktion und zum Recycling zu erarbeiten und zu verstehen, wie demografische und sozioökonomische Aspekte das Abfallverhalten beeinflussen. Die Ergebnisse dieser Arbeit sollten idealerweise die qualitativen Untersuchungen des BAFU ergänzen. Eine solche Erkenntnis wäre für Bund, Kanton und Gemeinde von hoher Relevanz und könnte dazu beitragen, Abfall- und Verbesserungsmassnahmen zielgerichtet zu planen und Steuermittel effektiver einzusetzen.

4.1 Cluster-Analyse

Vattani (2011)[15] beschreibt in seinem Paper die k-means-Methode als einen der am häufigsten genutzten Algorithmen im Bereich des geometrischen Clusterings. Dieser lokal operierende Algorithmus teilt n Datenpunkte in K disjunkte Teilmengen (Cluster) S_j, mit jeweils N_j Datenpunkten, auf. Dabei wird jedem Datenpunkt X_n das nächstgelegene Cluster-Zentrum μ_j zugeordnet und nach jeder Zuordnung der Mittelwert jedes Clusters S_j neu kalkuliert. Dieser Prozess wird solange wiederholt, bis das System stabil bleibt und das Clustering abgeschlossen ist.

$$J = \sum_{j=1}^{K} \sum_{n \in \mathcal{S}_j} \left| x_n - \mu_j \right|^2$$

Es gibt zahlreiche Cluster-Algorithmen, die sich sowohl im Distanzmass zwischen den Beobachtungen als auch in der Art der Input-Daten unterscheiden. Der Fokus dieses Abschnitts liegt auf dem k-means Algorithmus, der zur Untersuchung der Hypothese H1 verwendet wird: *"Es existieren verschiedene Gemeindegruppen, die sich hinsichtlich ihres Abfallaufkommens der Haushalte, sowie weiteren sozioökonomischen Faktoren differenzieren"*.
Im Clustering-Prozess dieser Untersuchung wird bewusst eine allgemeine Hypothese gewählt. Das Hauptziel besteht darin, Muster in den Daten zu erkennen, ohne durch eine zu präzise Hypothese eingeschränkt zu werden. Dies ermöglicht es, vielfältige und unerwartete Cluster-Formationen zu identifizieren, die im Kontext von Abfallproduktion und sozioökonomischen Faktoren der Gemeinden relevant sein könnten.

4.1.1 Vorbereitungen und Vorgehen

Zur Vermeidung von Verzerrungen im Clustering und basierend auf der explorativen Datenanalyse wurden folgende Schritte unternommen: Vier neue Features, die Kehrichtmenge, Altpapier, Altglas und Arbeitslosigkeit, wurden pro Gemeinde und Jahr per capita definiert, anstatt sie in Tonnen bzw. Personen zu aggregieren. Unplausible Daten, darunter 289 NaN-Zeilen und die im Kapitel 3 identifizierten Ausreisser bzw. potenzielle Datenfehler, wurden entfernt. Das bisherige Dataframe-Design, das pro Gemeinde und Jahr eine separate Zeile für den Indikator Value für jeden der 17 Datensätze aufwies, wurde geändert. Es wurde eine Pivot-Tabelle erstellt, die alle Beobachtungen pro Gemeinde und Jahr bündelt und die verschiedenen Werte für jeden Datensatz in getrennten Spalten darstellt. Dies reduziert die Anzahl der Zeilen auf 10'815 und ändert die Anzahl der Features entsprechend auf 23.

Für die Anwendung des k-means Algorithmus auf den vorhandenen Pivot-Datensatz sind weitere Vorarbeiten nötig. Die Voraussetzungen der Datenaufbereitung werden im Artikel von Ryzhkov (2020)[9] anschaulich aufgelistet und bilden die Grundlage für weitere Anpassungen.
Der k-means-Algorithmus analysiert die Distanz zwischen Datenpunkten und setzt daher voraus, dass die Inputvariablen numerisch sind. Kategorielle Variablen, beispielsweise das Beobachtungsjahr, können im k-means-Verfahren nicht sinnvoll verarbeitet werden.
Ein weiteres Hindernis im k-means Clustering ist die unterschiedliche Varianz der Features (vgl. Abbildung 4). Die Varianz eines Features misst dessen Wertstreuung und beeinflusst direkt sein Gewicht im Clustering-Prozess. Features mit hoher Varianz neigen daher dazu, den Clustering-Prozess zu dominieren. Um dieses Problem zu adressieren, wird der StandardScaler aus der Pandas-Bibliothek verwendet. Dadurch werden die Features so skaliert, dass sie eine einheitliche Varianz von 1 und einen Mittelwert von 0 aufweisen, wodurch ein gleichmässiger Einfluss im Clustering sichergestellt wird.
In der Vorbereitungsphase werden auch numerische Variablen auf Kollinearität untersucht, da, wie Ryzhkov 2020 ausführte, stark korrelierte Variablen für Clustering-Algorithmen problematisch sein

können. Sie erzeugen "Noise", ähnlich wie Ausreisser, die nach der explorativen Analyse bereits entfernt wurden. Variablen, die Bevölkerungsstruktur, politische Ausrichtung und Wohlstand in den Gemeinden des Kanton Zürichs repräsentieren, zeigen innerhalb ihrer jeweiligen Kategorien eine hohe Korrelation (vgl. Abbildung B im Anhang). Daher ist es sinnvoll, sich in der Clusteranalyse auf jeweils eine Variable pro Kategorie zu konzentrieren. Als Eingangsvariablen für die k-means Clusteranalyse werden folglich ausgewählt: *"Arbeitslosigkeit per capita", "durchschnittliches Einkommen", "Nettoaufwand Umweltschutz der Gemeinde", "Haushaltabfall/Kehricht per capita", "NRW Wähleranteil SVP" und "Anzahl Studierende auf 1000 Personen".*

Die Auswahl der sechs Variablen berücksichtigt auch die Datenverfügbarkeit des Abfalles über den Zeitraum von 1996 bis 2021. Würde etwa der Wasserverbrauch einbezogen, wäre dies problematisch, da für diese Variable nur Daten ab 2006 vorhanden sind. In der Pivot-Datentabelle würde dies zu "NaN"-Werten führen, die vom k-means Algorithmus nicht verarbeitet werden können. Diese Herausforderung wird auch in der Arbeit von Chi, Chi, und Baraniuk (2016)[5] hervorgehoben. Dort wird erläutert, dass Clustering-Methoden vollständige Datensätze benötigen und für unvollständige Daten im Allgemeinen zwei Strategien existieren: Löschung oder Imputation.
Für die Variable des Wähleranteils in den Nationalratswahlen, die alle vier Jahre stattfinden, wird die Imputationsstrategie angewendet. Hier wird angenommen, dass die Wähleranteile über eine vierjährige Amtsperiode konstant bleiben. Das ermöglicht die Vervollständigung des Datensatzes und die Anwendung der k-means Clusteranalyse.

Schliesslich wird die Thematik der Dimensionalität etwas genauer betrachtet. Wie Singh (2021)[12] in einem Online-Artikel verdeutlicht, neigen die Abstände zwischen verschiedenen Datenpunkten dazu, nahe beieinander zu liegen wenn die Dimension zunimmt. Dies ist insofern ein Problem, da Algorithmen wie k-means abstandsbasierte Metriken verwenden. Dieses Problem ist auch als "Curse of dimensionality" bekannt. Dieser Begriff wurde von Bellman im Jahre 1957 eingeführt. In der Arbeit von Venkat (2018)[16] wird zudem die weiteren Implikationen von erhöhter Dimensionalität deutlich: exponentieller Anstieg des Berechnungs- sowie Speicheraufwandes und insbesondere auch: eingeschränkte Visualisierungs- und entsprechend Interpretationsmöglichkeiten.
Um diesen Problemen entgegenzuwirken, wird eine Reduktion der Dimensionalität durch Principal Component Analysis (PCA) vorgenommen. Gewers et al. (2018)[7] verdeutlichen das Potenzial von PCA in der multivariaten Statistik und dessen Fähigkeit, Dimensionen zu reduzieren. Bei der PCA-Implementierung wird davon ausgegangen, dass Features mit höherer Varianz informativer sind. Daher ist es entscheidend, die Anzahl der zu behaltenden PCA-Features sorgfältig zu wählen.
Nach dieser Vorbereitung wird die k-means-Clusteranalyse auf den Datensatz angewendet. Dieser Prozess wird zweimal durchgeführt, um die Auswirkungen unterschiedlicher Dimensionen zu untersuchen.

Bei der ersten Durchführung wird ein dreidimensionaler Datenraum genutzt, indem die Anzahl der PCA-Komponenten auf drei beschränkt wird. Anschliessend wird die Analyse erneut durchgeführt, diesmal jedoch in einem reduzierten zweidimensionalen Raum. Dieser Vergleich ermöglicht ein tieferes Verständnis, wie die Dimensionalität der Daten die Ergebnisse der Clusteranalyse beeinflusst. Für Vergleichszwecke wird ebenfalls ein hierarchisches Clustering auf dem Datensatz angewendet. Laut Abdalla (2022)[1] wählt diese Methode zufällige Hauptpunkte für jedes Cluster und weist dann iterativ Datenpunkte Clustern zu, ähnlich wie bei k-means. Der Unterschied liegt in der Zielsetzung: Hierarchisches Clustering versucht, eine Cluster-Hierarchie aufzubauen, die durch ein Dendrogramm visualisiert werden kann. Obwohl k-means und hierarchisches Clustering ähnliche Leistungstrends aufweisen können, sind Abweichungen je nach Datensatz möglich.

4.1.2 Anwendung und Ergebnisse Clustering

In diesem Analyseprozess wird eine Pipeline der Sklearn-Bibliothek genutzt, um verschiedene Variablen für eine Cluster-Analyse vorzubereiten. Diese Variablen, wie im vorherigen Abschnitt vorgestellt, werden zunächst mithilfe des StandardScalers transformiert. Diese Transformation stellt sicher, dass alle Verteilungen einen Mittelwert von 0 und eine Standardabweichung von 1 aufweisen, was eine einheitliche Analyse der unterschiedlichen Variablen ermöglicht.

Im nächsten Schritt wird mithilfe einer Hauptkomponentenanalyse (PCA) bestimmt, wie viel der Gesamtvarianz im Datensatz von jeder Hauptkomponente erklärt wird (vgl. Abbildung C im Anhang). Die Ergebnisse dieser Analyse zeigen, dass die ersten beiden PCA-Hauptkomponenten (jeweils als Linearkombination der ursprünglichen Variablen) zusammen etwa 50 % der Gesamtvarianz im Datensatz erklären, während die Kombination der ersten drei Hauptkomponenten bis zu 66 % der Gesamtvarianz abdeckt. Diese Anteile erscheinen für die geplante Analyse ausreichend.

Der k-means Clustering-Algorithmus wird zunächst auf einen dreidimensionalen Datensatz angewendet. Hierfür werden eine StandardScaler-Instanz und eine PCA-Instanz mit drei Komponenten erstellt, die in einer Pipeline kombiniert und auf die Daten angewendet werden. In der Folge wird für eine Reihe unterschiedlicher Anzahlen an Clustern (von 2 bis 10) ein k-means Clustering durchgeführt. Für jedes dieser Modelle wird der Silhouette Score berechnet und gespeichert. Nachdem das Modell mit dem höchsten Silhouette Score ermittelt wurde, wird das k-means Clustering erneut mit der optimalen Anzahl von Clustern nach Abgleich Silhouette Score durchgeführt. Dies repräsentiert eine einzige Durchführung des Algorithmus.
Ein bekanntes Problem bei k-means besteht darin, dass das Endergebnis stark von den initial gewählten Zentren abhängt, die zufällig ausgewählt werden. Aufgrund dieser Variation in den

Clustering-Ergebnissen durch die zufällige Initialisierung von k-means, wird das gesamte Clustering 10 weitere Male ausgeführt. Dabei wird jedes Mal erneut das Modell mit dem höchsten Silhouette Score oder dem geringsten "Sum of Squares Error" (SSE) ermittelt. Die Ausführung von k-means, die den Silhouette Score maximiert, wird ausgewählt, da ein hoher Score darauf hinweist, dass die Datenpunkte innerhalb der Cluster nahe beieinander liegen. Eine detaillierte Diskussion über die Validierung von Clustering mittels Silhouette Score und SSE wird von Thinsungnoen et al. (2015)[14] präsentiert.
Nachdem das k-means Clustering für 2 bis 10 Cluster jeweils 10 Mal ausgeführt wurde, wird das beste Modell ausgewählt und evaluiert. Da bei der Anwendung der PCA ein dreidimensionaler Raum betrachtet wird, ist die Visualisierung des "besten" Clustering in Form eines 3D-Plots möglich (Abbildung 6).

Die folgenden Ergebnisse können bei wiederholter Ausführung des Programmcodes im Anhang geringfügige Abweichungen aufweisen. Die grundlegende Struktur und die Kernergebnisse bleiben jedoch über alle Durchläufe hinweg konsistent:
Innerhalb des dreidimensionalen Raums ergibt die Anwendung von K-means mit $n = 4$ Clustern den höchsten Silhouette-Score von ca. 0.28 (Abbildung 5). Der Silhouette-Score dient als Mass für die Qualität einer Clusterbildung und variiert zwischen -1 und 1. Ein hoher Silhouette-Score weist darauf hin, dass die Objekte innerhalb ihrer Cluster gut zueinander passen und sich gut von anderen Clustern abgrenzen lassen, wie in der von Thinsungnoen et al. (2015) zitierten Arbeit dargelegt.
Ein Silhouette-Score von 0.28 deutet auf eine mittelmässige Clusterstruktur hin: Die Objekte innerhalb der einzelnen Cluster sind weder besonders dicht beieinander, noch sind die Cluster besonders gut voneinander abgegrenzt. Zwar weist dieser Wert keine ausgeprägte Clusterstruktur auf, doch bedeutet er auch nicht zwangsläufig eine unzureichende Struktur.

Abbildung 5: Silhouette Scores

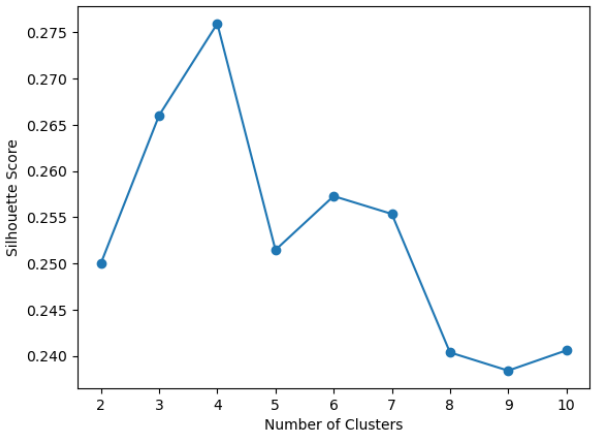
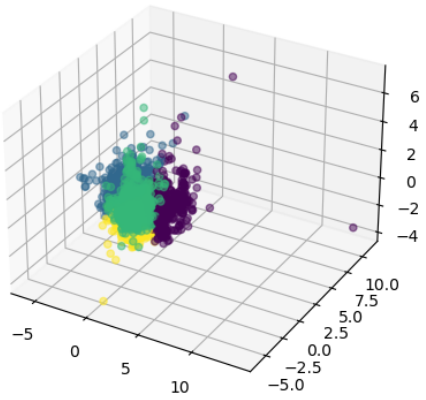


Abbildung 6: Visualisierung Clustering im 3D-Raum



Im zweidimensionalen Raum wird mittels K-means und $n = 3$ Clustern ein deutlich höherer Silhouette-Score von ca. 0.45 erzielt. Die Methode der Hauptkomponentenanalyse (PCA) wird typischerweise genutzt, um die Dimensionalität von Daten zu reduzieren und dabei gleichzeitig möglichst viel von der in den Daten vorhandenen Variabilität zu bewahren. In diesem Fall scheint die Reduzierung auf zwei Dimensionen zu einer klareren Datenstruktur geführt zu haben: Potenzielles "Rauschen" in den Daten wurde minimiert, was es ermöglicht, die tatsächliche Struktur der Daten deutlicher hervorzuheben. Allerdings ist es wichtig zu beachten, dass die Reduzierung von Dimensionen auch Informationen verlieren lässt, was zu einer übermässigen Vereinfachung der tatsächlichen Datenstruktur führen kann.

Abbildung 7: Silhouette Scores

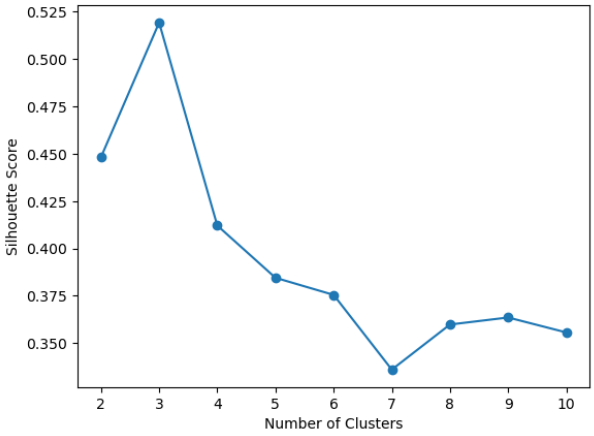
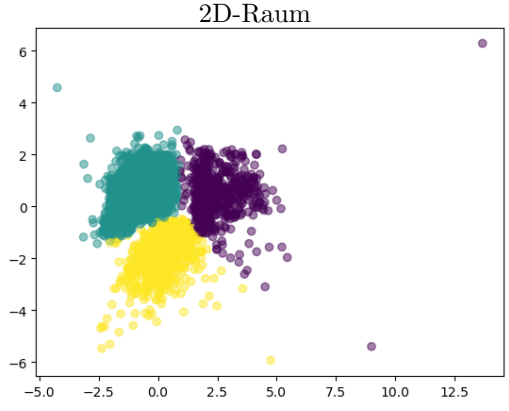


Abbildung 8: Visualisierung Clustering im 2D-Raum

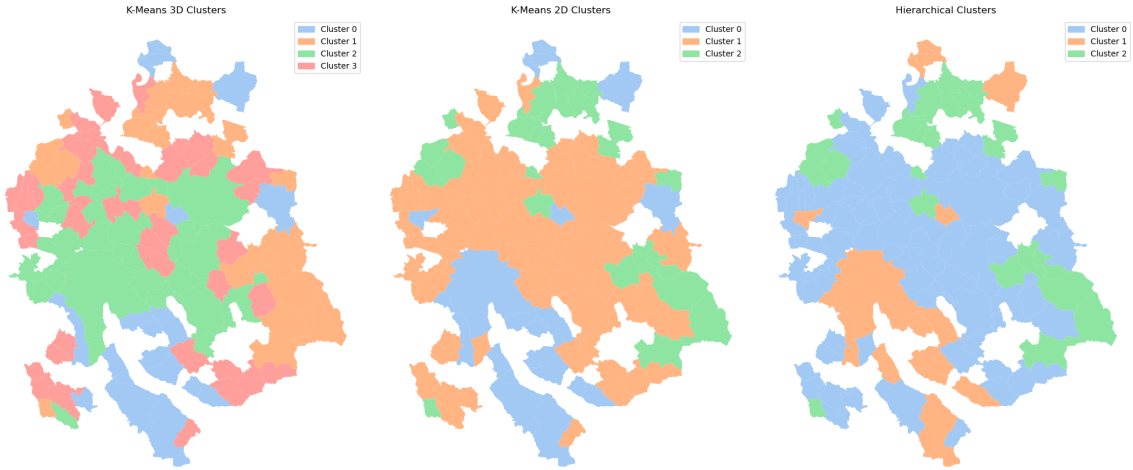


Bei der Anwendung des hierarchischen Clustering-Verfahrens stellt sich heraus, dass erneut mit $n = 4$ Clustern das "beste" Clustering erreicht wird. Die dabei erzielten Silhouette-Scores liegen in einem Bereich zwischen 0.40 und 0.55. Es ist zu beobachten, dass die Scores bei verschiedenen Durchläufen leicht variieren (vgl. Abbildungen D, E im Anhang).

Abschliessend wurden die Cluster-Ergebnisse der einzelnen Gemeinden - abhängig vom verwendeten Modell und den entsprechenden Spezifikationen - erneut in den Datensatz zurückgeführt und die Regionen entsprechend ihrem Zugehörigkeitscluster eingefärbt (Abbildung 9). Mit anderen Worten, die Cluster, die auschliesslich auf Basis von sozioökonomischen und Abfalldaten erstellt wurden, wurden in einen geografischen Kontext gesetzt. Nach der Cluster-Analyse geht hervor, dass eine Gemeinde von Jahr zu Jahr einem anderen Cluster angehören kann oder nach einem bestimmten Jahr das Cluster wechselt. Für die grafische Darstellung wurde hingegen die Annahme getroffen, dass eine Region nur zu einem Cluster gehören kann. Zu diesem Zweck wurde eine Funktion erstellt, die für jedes Gebiet das über die Jahre dominierende Cluster - also das Cluster mit der meisten Zugehörigkeit innerhalb einer Gemeinde - ermittelt. Die geografische Darstellung der Cluster wirkt über die verschiedenen Clustering-Methoden hinweg bemerkenswert ähnlich. Es scheint, dass die Stadt Zürich und angrenzende Gemeinden im Zentrum des Kantons ein eigenes Cluster bilden. Ebenso zeigt der Norden/Osten des

Kantons eine andere Struktur in den Daten, ebenso wie der Süden bzw. Südwesten. Allerdings variieren diese Einteilungen je nach gewählter Methode und angewendeter Dimensionsreduktion.

Abbildung 9: Geografische Visualisierung der Ergebnisse der Clusterverfahren



Mit den nun gewonnen Erkenntnissen wird der zweite Teil der Data-Mining-Analyse im Rahmen dieser Studienarbeit angegangen: Die Regressions-Analyse.

4.2 Regressions-Analyse mit Paneldaten

Zur Ergänzung der durchgeführten Cluster-Analyse könnte das Interesse bestehen, die einflussreichsten Faktoren für das Abfallaufkommen in Zürich und den verschiedenen Clustern zu ermitteln. Das Regressionsverfahren bietet sich hierfür an. Ziel der Regressionsanalyse ist es, die Beziehung zwischen einer abhängigen und einer oder mehreren unabhängigen Variablen zu modellieren.

4.2.1 Vorbereitungen und Vorgehen

In der Regressionsanalyse liegt der Fokus auf den Variablen des Datensatzes, die voraussichtlich einen direkten oder indirekten Einfluss auf das Abfallaufkommen ausüben. Im Kontext dieser Studie könnte die zweite Hypothese daher wie folgt formuliert werden:
Das Abfallaufkommen der Haushalte in einer Gemeinde wird durch Faktoren wie Arbeitslosigkeit, Ausgaben für den Umweltschutz pro Kopf, Bildungsgrad und die politischen Präferenzen beeinflusst.

Für aussagekräftige kausale Schlüsse aus Regressionsanalysen ist die Berücksichtigung von Störvariablen entscheidend. Manchmal können solche Variablen jedoch nicht in das Modell integriert werden, etwa weil sie nicht messbar sind. Nicht berücksichtigte Störvariablen können das Modell verzerren und verdeutlichen, warum Korrelation nicht gleich Kausalität ist. Detailliertere Erklärungen zu den Eigenschaften und Auswirkungen von Störvariablen finden sich in der Arbeit von Skelly, Dettori und Brodt (2012)[13].

In den Panel-Daten des Kanton Zürich sind zwei Arten von Störfaktoren relevant:
Einheitsfixe Störfaktoren beziehen sich auf Eigenschaften der Einheiten (Gemeinden), die zeitlich konstant bleiben. So wird angenommen, dass etwa die Einstellung zum Recycling innerhalb einer Gemeinde über die Zeit stabil bleibt. Ähnlich könnten die Anzahl der Recycling-Stationen oder die Verrechnungsweise des Hausmülls als einheitsfixe Variablen betrachtet werden.
Zeitfixe Störfaktoren betreffen Effekte, die alle Einheiten jährlich gleich beeinflussen, jedoch im Datensatz nicht erfasst sind. Beispiele hierfür könnten das jährliche Wetter oder die allgemeine Stimmung und Sorgen der Bevölkerung sein, die in allen Gemeinden ähnlich wirken.

Zur Berücksichtigung dieser nicht messbaren Störfaktoren wird ein Two-Way Fixed Effects (2FE) Regressionsmodell auf Panel-Daten angewendet. Das Modell orientiert sich an der Darstellung von Imai und Kim (2019)[8], ergänzt durch zeitfixe Effekte für die betrachtete Zeitperiode:

$$Y_{it} = \alpha_i + \lambda_t + \beta X_{it} + \epsilon_{it}$$

In der dargestellten Formel repräsentiert Y_{it} die abhängige Variable für die Einheit i in Periode t, α_i den einheitsfixen Effekt für Einheit i, λ_t den zeitfixen Effekt für Zeitperiode t, X_{it} die unabhängigen Variablen für Einheit i in Periode t mit den jeweiligen Koeffizienten β sowie Fehlerterm ϵ_{it}.

Folgende Transformationen können sowohl individuelle als auch periodenspezifische Effekte aus dem Modell eliminieren und sind Grundlage für die entsprechenden Python-Bibliotheken, welche dies im Rahmen eines 2FE Modells im Hintergrund ausführen:

$$\bar{y}_i = \frac{1}{T}\sum_t^T y_{it}, \quad \bar{y}_t = \frac{1}{N}\sum_i^N y_{it}, \quad \bar{y} = \frac{1}{NT}\sum_t^T \sum_i^N y_{it}$$

$$(y_{it} - \bar{y}_i - \bar{y}_t + \bar{y}_{it}) = \beta' (x_{it} - \bar{x}_i - \bar{x}_t + \bar{x}_{it}) + (u_{it} - \bar{u}_i - \bar{u}_t + \bar{u}_{it})$$

4.2.2 Anwendung und Ergebnisse von 2FE

Der reduzierte Datensatz in Pivot-Darstellung, der bereits für die Cluster-Analyse verwendet wurde, dient auch als Grundlage für das 2FE Modell. Zusätzlich werden die Gemeindenamen (GEBIET _ NAME), Jahre (INDIKATOR _ JAHR) sowie drei neue Spalten für Clusterzugehörigkeiten ergänzt. Diese Cluster-Spalten werden als Dummy-Variablen in das lineare Modell integriert.

Das untersuchte Modell ist ein Panel-Daten-Modell, das nach der Methodologie des Kapitels 4.2.1 erstellt wurde. Es wurde mit der PanelOLS-Methode aus dem linearmodels Python-Paket implementiert. Das Modell zielt darauf ab, Hypothese H2 zu testen, indem es den Einfluss verschiedener unabhängiger

Variablen auf die Menge an Abfall in Tonnen pro Kopf untersucht. Folgende Variablen und Faktoren werden in die Analyse einbezogen:

- Arbeitslosigkeit pro Kopf: Misst den Einfluss der Arbeitslosenrate.
- Wähleranteil der SVP bei den Kantonsratswahlen: Berücksichtigt politische Neigungen.
- Ausgaben für den Umweltschutz pro Kopf: Logarithmisch transformiert, um nicht-lineare Beziehungen abzubilden.
- Steuerpflichtiges Einkommen: Ebenfalls logarithmisch transformiert.
- Anzahl der Studierenden pro 1'000 Einwohner: Misst den Einfluss des Bildungsniveaus.
- Interaktionsterm zwischen Arbeitslosigkeit und Umweltausgaben pro Kopf: Erfasst den kombinierten Effekt dieser beiden Variablen auf die Abfallmenge.
- Clusterzugehörigkeit: Die Datenpunkte werden bestimmten Clustern zugeordnet.
- Einheitseffekte: Nicht beobachtete, zeitkonstante Unterschiede zwischen den Einheiten.
- Zeiteffekte: Nicht beobachtete Faktoren, die sich über die Zeit ändern und alle Einheiten gleich beeinflussen.

Das Hinzufügen von Variablen wie dem Altpapier- und Altglas-Aufkommen pro Person könnte zwar intuitiv erscheinen, aber es gibt gute Gründe, diese aus dem Modell auszulassen. Erstens könnte die Einbeziehung dieser Variablen zu Endogenitätsproblemen führen, da das Recyclingverhalten der Bevölkerung sowohl Einfluss auf diese Variablen als auch auf die abhängige Variable (Abfallmenge pro Kopf) haben könnte. Zweitens könnten diese Variablen auch eine gewisse Multikollinearität ins Modell bringen.

Tabelle 1 präsentiert die geschätzten Parameter nach Anwendung des 2FE Modells auf den Datensatz. Während einige statistische Ausschläge bezüglich der abhängigen Variablen festzustellen sind, muss auch die praktische Relevanz dieser Ergebnisse evaluiert werden. Die Ergebnisse des linearen Modells weisen auf folgende Beobachtungen hin:
Der Koeffizient für Arbeitslosigkeit pro Kopf beträgt -3.5491, was bedeutet, dass eine Zunahme der Arbeitslosigkeit pro Kopf um eine Einheit mit einer Abnahme der Kehrichtmenge pro Kopf um 3.5491 Einheiten verbunden ist, bei konstant gehaltenen anderen Faktoren. Dieser Effekt ist mit einem p-Wert von 0.0012 statistisch signifikant auf dem 5 %-Niveau.
Eine 1 %-ige Erhöhung der erwarteten Umweltausgaben pro Kopf in einer Gemeinde ist mit einer Abnahme der Kehrichtmenge um 0.0144 Einheiten assoziiert, was ebenfalls auf dem 5 %-Niveau signifikant ist ($p < 0{,}05$).

Tabelle 1: Ergebnisse der Regressionsanalyse von Paneldaten unter Verwendung eines 2FE-Modells

	Parameter	Std. Err.	T-stat	P-value	Lower CI	Upper CI
unemployment_capita	-3.5491	1.0950	-3.2411	0.0012	-5.6960	-1.4021
nrw_svp	-4.09e-05	0.0003	-0.1274	0.8986	-0.0007	0.0006
np.log(exp_environmental_protection_capita)	-0.0144	0.0029	-5.0236	0.0000	-0.0200	-0.0088
np.log(taxable_income_natural)	-0.0121	0.0126	-0.9586	0.3378	-0.0367	0.0126
students_per_1000pop	-0.0001	0.0001	-1.1610	0.2457	-0.0003	8.013e-05
cluster_kmeans_2D_1	-0.1296	0.0522	-2.4808	0.0132	-0.2320	-0.0272
cluster_kmeans_2D_2	-0.1144	0.0482	-2.3716	0.0178	-0.2090	-0.0198
cluster_kmeans_3D_1	0.0010	0.0118	0.0868	0.9309	-0.0221	0.0242
cluster_kmeans_3D_2	0.0261	0.0029	9.1174	0.0000	0.0205	0.0317
cluster_hierarchical_1	-0.1159	0.0514	-2.2522	0.0244	-0.2167	-0.0150
cluster_hierarchical_2	0.0139	0.0182	0.7623	0.4459	-0.0218	0.0495
unemployment_capita: np.log(exp_environmental_protection_capita)	0.7579	0.2244	3.3769	0.0007	0.3179	1.1979

Die Zuordnung zu Cluster 2 über eine k-means-Analyse im zweidimensionalen Raum zeigt einen negativen und signifikanten Zusammenhang auf dem 5 %-Niveau: Die Zugehörigkeit zu diesem Cluster ist mit einem Rückgang der Kehrichtmenge um 0.1144 Einheiten im Vergleich zum Referenzcluster assoziiert.

Interessanterweise weist der Interaktionsterm zwischen Arbeitslosigkeit und erwarteten Umweltausgaben einen positiven Koeffizienten von 0.7579 auf. Dies deutet darauf hin, dass der kombinierte Effekt dieser beiden Variablen auf die Kehrichtmenge sich von ihren individuellen Effekten unterscheidet. Genauer gesagt, schwächt eine Zunahme der Arbeitslosigkeit pro Kopf den negativen Effekt der Umweltausgaben auf die Kehrichtmenge ab, was auf dem 5 %-Niveau signifikant ist.

Der R-Quadrat-Wert von 0.219 deutet darauf hin, dass die Prädiktoren im Modell etwa 21.9 % der Variabilität in der Kehrichtmenge pro Kopf erklären. Der RMSE von 0.019 reflektiert den durchschnittlichen Unterschied zwischen den beobachteten und den vom Modell vorhergesagten Werten.

Die im Anhang aufgeführten Programmcodes testen die formalen Annahmen der Regression. Es wird angenommen, dass die Parameter linear sind (d.h., dass ein additives lineares Regressionsmodell vorliegt) und dass keine Autokorrelation in den Fehlertermen besteht, d.h. die Fehlerterme sind unabhängig. Des Weiteren wird die Homoskedastizität getestet und analysiert, ob die Fehlerterme normalverteilt sind. Eine erneute Überprüfung der Multikollinearität der Prädiktoren wurde nicht vorgenommen, da dies bereits im Rahmen der Cluster-Analyse erfolgt ist.

5 Fazit

In dieser Studienarbeit wurde eine kombinierte Methode aus Clustering und Regression angewendet, um die Faktoren zu identifizieren, die das Kehricht-Abfallaufkommen im Kanton Zürich beeinflussen. Der k-means-Algorithmus wurde zur Clusterbildung und ein Two-Way-Fixed-Effects (2FE) Regressionsmodell zur Analyse der einflussreichsten Variablen verwendet.
Die Datenvorbereitung spielte dabei eine entscheidende Rolle: Ausreisser wurden entfernt, fehlende Daten imputiert, Features hinsichtlich Skalierung und Varianz sorgfältig ausgewählt und transformiert. Zudem wurde das Problem der hohen Dimensionalität durch Methoden der PCA minimiert.

Im Bereich des Clusterings bildeten die Stadt Zürich und angrenzende Gemeinden ein eigenes Cluster. Die Silhouette-Scores lagen zwischen 0.40-0.55. Bei der Regresssionsanalyse der Panel-Daten zeigte sich, dass Arbeitslosigkeit und Ausgaben der jeweiligen Gemeinden für Umweltschutz und Raumplanung pro Kopf einen negativen Einfluss auf das Abfallaufkommen haben, während politische Präferenzen und Bildungsniveau in dem hier betrachteten Modell keinen signifikanten Einfluss zeigten.

Mit den vorliegenden Daten lassen sich jedoch keine Bewertung über die Qualität des Kehrichts treffen. Die Ergebnisse aus den Modellen sollten als ergänzende Informationen zu den qualitativen Untersuchungen des BAFU angesehen werden. Ein langfristiger Zugang zu den qualitativen Daten des Bundesamtes wäre wünschenswert. Durch den freien Zugang auf diese Daten könnten neue Datenpunkte in die Modelle integriert und deren Qualität verbessert werden.
Trotzdem können die Resultate der Analyse das Potenzial von Data Mining Projekten öffentlicher Daten aufzeigen. Es empfiehlt sich, solche Vorhaben häufiger zu realisieren und ihre Ergebnisse zu veröffentlichen. Sie könnten als zusätzliche Basis dienen, um strategische Ziele zur Optimierung der Kehrichtsituation im Kanton Zürich festzulegen.

Diese Erkenntnisse unterstreichen die Bedeutung einer sorgfältigen Datenvorbereitung und -analyse und bieten nützliche Richtlinien für zukünftige Projekte, die ähnliche Methoden und Datensätze verwenden möchten. Insbesondere zeigen sie, wie die Kombination von unüberwachtem und überwachtem Lernen tiefergehende Einblicke in komplexe Fragestellungen ermöglichen kann.

6 Quellenverzeichnis

[1] Abdalla, Hassan I. 2022. "A Brief Comparison of K-means and Agglomerative Hierarchical Clustering Algorithms on Small Datasets." In Proceeding of 2021 International Conference on Wireless Communications, Networking and Applications, edited by Zhihong Qian, M.A. Jabbar, and Xiaolong Li, 623–32. Lecture Notes in Electrical Engineering. Singapore: Springer Nature. https://doi.org/10.1007/978-981-19-2456-9_64.

[2] Angrist, Joshua David, and Jörn-Steffen Pischke. 2009. *Mostly Harmless Econometrics: An Empiricist's Companion.* Princeton: Princeton University Press.

[3] Bundesamt für Umwelt BAFU. 2022. "Medienmitteilung: Zusammensetzung Des Kehrichts - BAFU Hat Neue Analyse Gestartet." November 15, 2022. https://www.admin.ch/gov/de/start/dokumentation/medienmitteilungen.msg-id-91533.html.

[4] Bundesamt für Umwelt BAFU and Steiger, Urs. 2012. "Erhebung Der Kehrichtzusammensetzung 2012." https://www.admin.ch/gov/de/start/dokumentation/medienmitteilungen.msg-id-51815.html.

[5] Chi, Jocelyn T., Eric C. Chi, and Richard G. Baraniuk. 2016. "k-POD: A Method for k-Means Clustering of Missing Data." *The American Statistician* 70 (1): 91–99. https://doi.org/10.1080/00031305.2015.1086685.

[6] Fayyad, Usama, Gregory Piatetsky-Shapiro, and Padhraic Smyth. 1996. "From Data Mining to Knowledge Discovery in Databases." AI Magazine 17 (3): 37–37. https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/1230.

[7] Gewers, Felipe L., Gustavo R. Ferreira, Henrique F. de Arruda, Filipi N. Silva, Cesar H. Comin, Diego R. Amancio, and Luciano da F. Costa. 2018. "Principal Component Analysis: A Natural Approach to Data Exploration." *ACM Computing Surveys* 54 (4): 1–34. https://doi.org/10.1145/3447755.

[8] Imai, Kosuke, and In Song Kim. 2019. "When Should We Use Unit Fixed Effects Regression Models for Causal Inference with Longitudinal Data?" *American Journal of Political Science* 63 (2): 467–90. https://doi.org/10.1111/ajps.12417.

[9] Ryzhkov, Evgeniy. 2020. "5 Stages of Data Preprocessing for K-means Clustering." *Medium (blog).* https://shorturl.at/NU127.

[10] "Schweizer:innen produzieren 703 kg Siedlungsabfälle pro Einwohner:in und Jahr." 2021. *Swiss Recycling*. September 16, 2021. https://swissrecycling.ch/de/aktuell/detail/abfallaufkommen-in-der-schweiz-und-europa.

[11] Shearer, Colin. 2000. "The CRISP-DM Model: The New Blueprint for Data Mining." *Journal of Data Warehousing* 5: 13-22.

[12] Singh, Shivangi. 2021. "K Means Clustering on High Dimensional Data." *The Startup (blog)*. https://medium.com/swlh/k-means-clustering-on-high-dimensional-data-d2151e1a4240.

[13] Skelly, Andrea C., Joseph R. Dettori, and Erika D. Brodt. 2012. "Assessing Bias: The Importance of Considering Confounding." *Evidence-Based Spine-Care Journal* 3 (1): 9–12. https://doi.org/10.1055/s-0031-1298595.

[14] Thinsungnoen, Tippaya, Nuntawut Kaoungku, Pongsakorn Durongdumronchai, Kittisak Kerdprasop, and Nittaya Kerdprasop. 2015. "The Clustering Validity with Silhouette and Sum of Squared Errors." https://doi.org/10.12792/iciae2015.012.

[15] Vattani, Andrea. 2011. "K-means Requires Exponentially Many Iterations Even in the Plane." Discrete & Computational Geometry. 45 (4): 596–616. https://doi.org/10.1007/s00454-011-9340-1.

[16] Venkat, Naveen. 2018. "The Curse of Dimensionality: Inside Out." https://doi.org/10.13140/RG.2.2.29631.36006.

A Appendix

A.1 Abbildungen und Code

Abbildung A: Fehlende Dateneinträge je Gemeinde

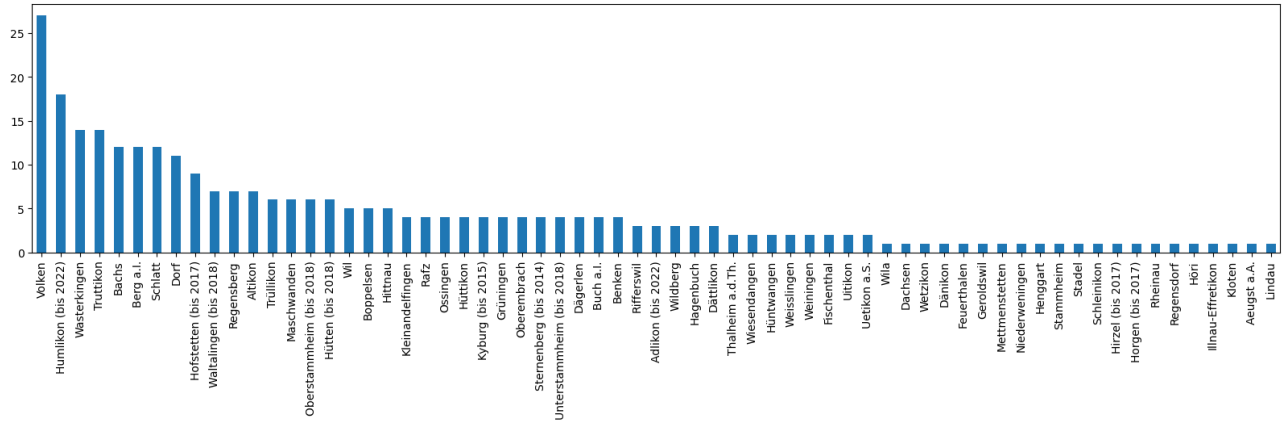


Abbildung B: Korrelation zwischen verschiedenen Variablen innerhalb des DataFrames

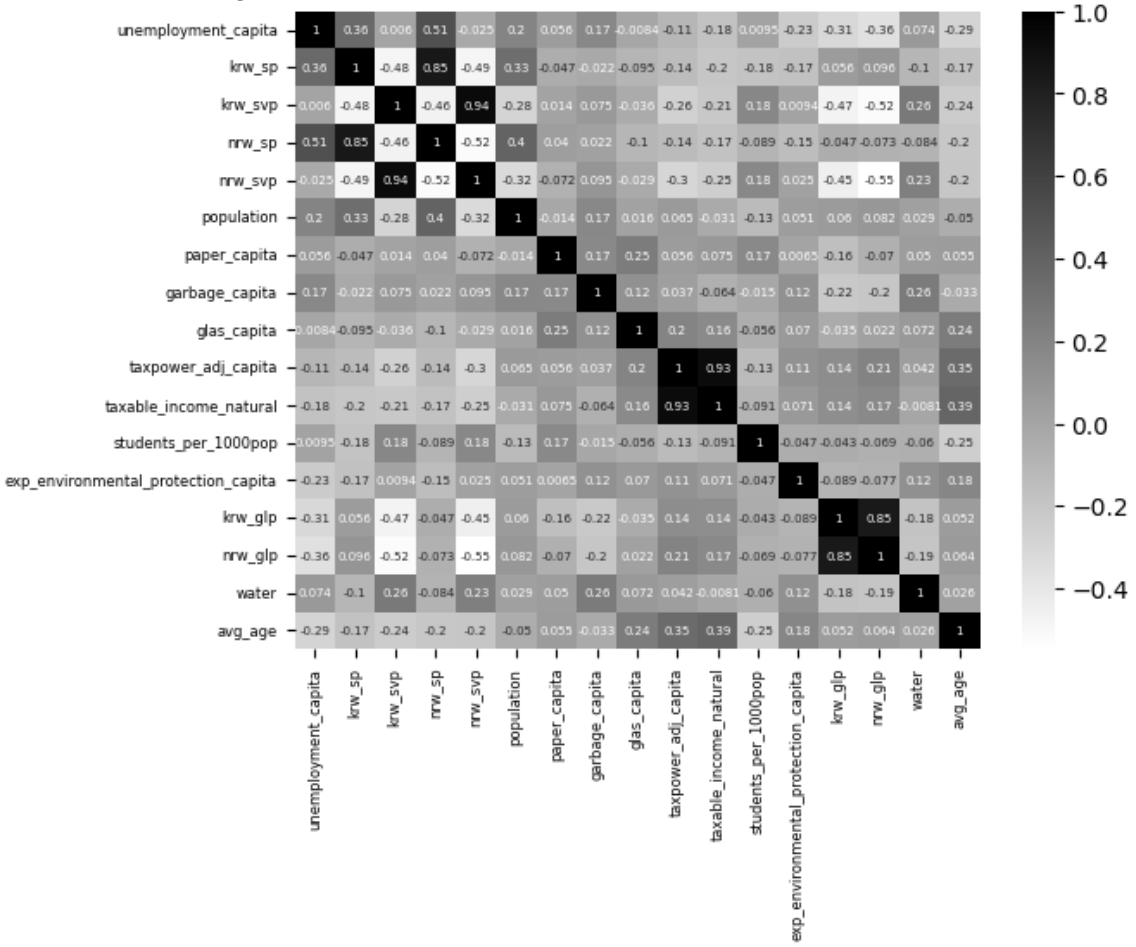


Abbildung C: Hauptkomponentenanalyse

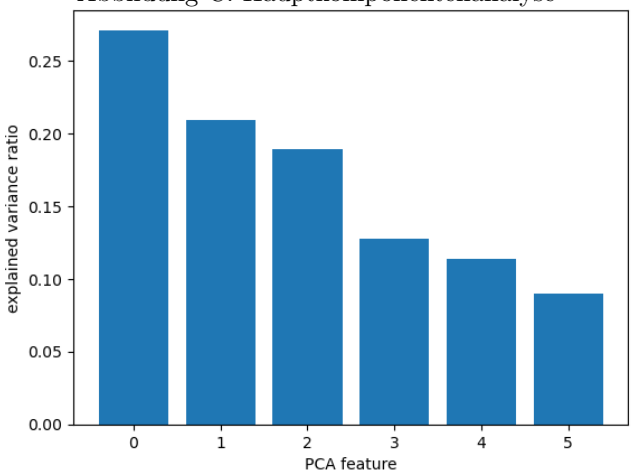


Abbildung D: Silhouette Scores für verschiedene Anzahl von Clustern

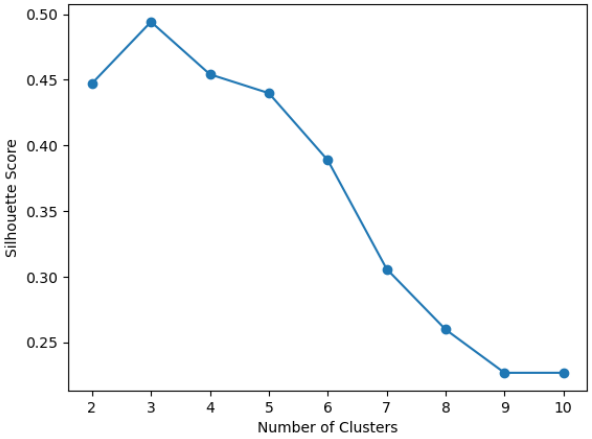


Abbildung E: Visualisierung Hierarchisches Clustering

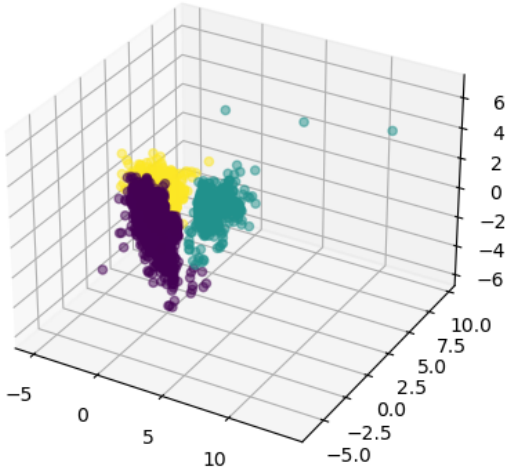


Abbildung F: Pairplot ausgewählter Features (k-Means, 2D)

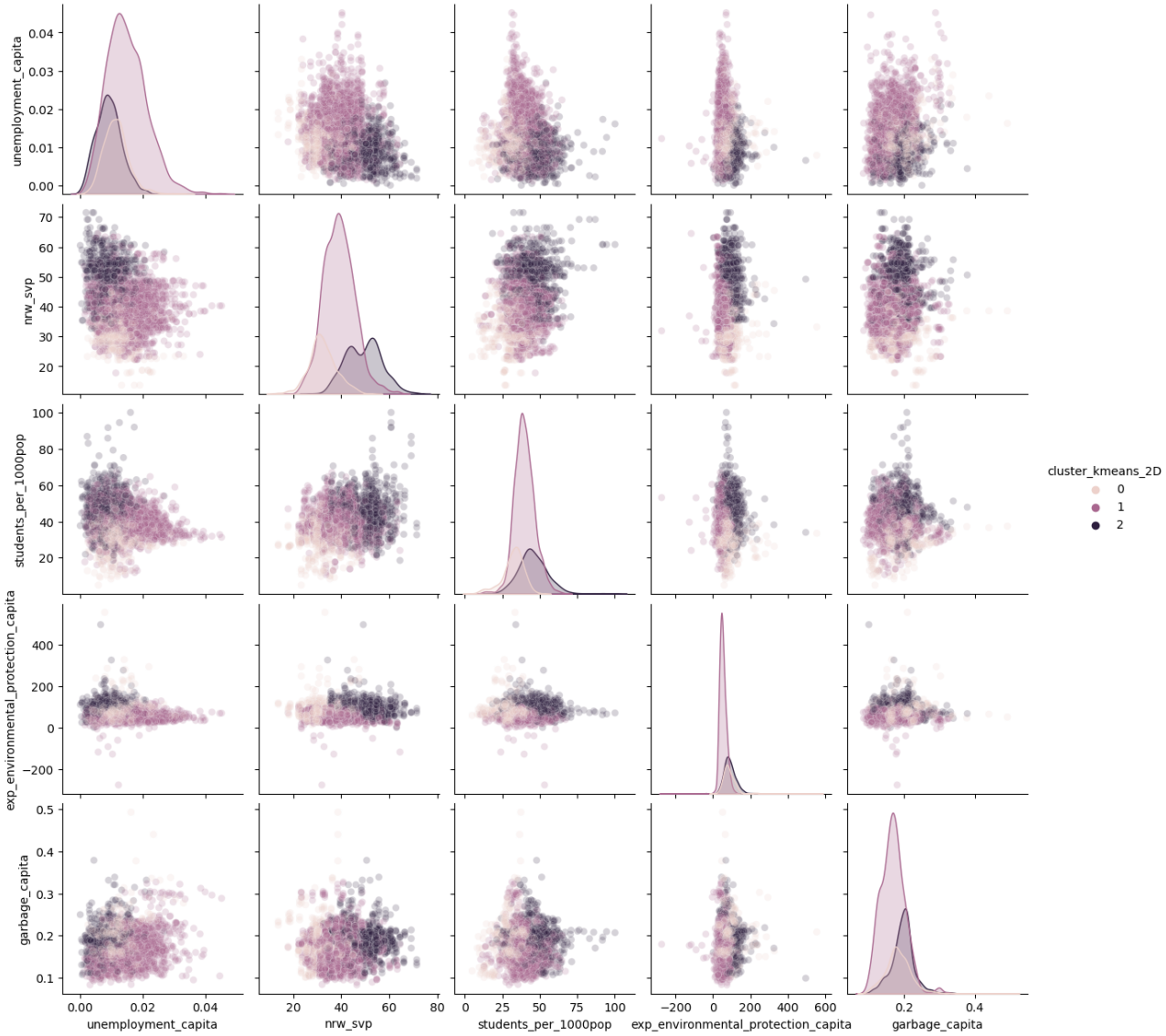


Abbildung G: Residuals vs. Fitted Values

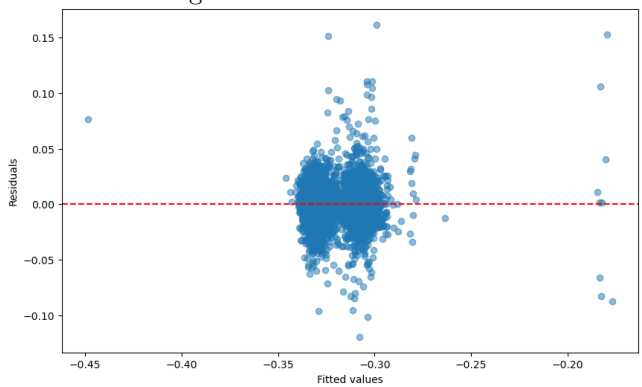


Abbildung H: QQ Plot of Residuals

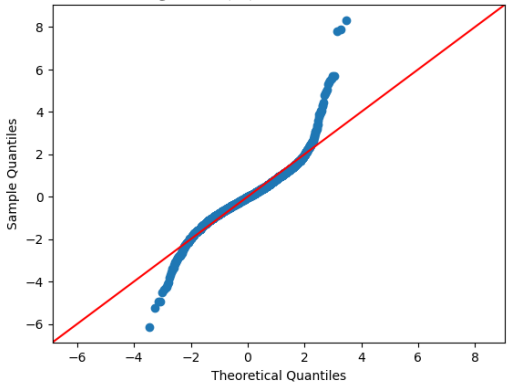


Abbildung I: Übersicht über die einzenen Datensätze

KRW Wähleranteil GLP [%]	Prozent	Statistisches Amt des Kantons Zürich	Wähleranteil GLP bei Kantonsratswahlen
NRW Wähleranteil GLP [%]	Prozent	Statistisches Amt des Kantons Zürich	Wähleranteil GLP bei Nationalratswahlen. Hinweis: 2015 werden alle Auslandschweizer zur Stadt Zürich gezählt.
Arbeitslose [Anz.]	Anzahl	Staatssekretariat für Wirtschaft (seco)	Jahresmittel der beim kantonalen Arbeitsamt eingeschriebenen Arbeitslosen. Aus Datenschutzgründen dürfen Werte von Gemeinden mit weniger als 6 Arbeitslosen (Jahresmittel) nicht ausgewiesen werden.
KRW Wähleranteil SP [%]	Prozent	Statistisches Amt des Kantons Zürich	Wähleranteil SP bei Kantonsratswahlen
NRW Wähleranteil SP [%]	Prozent	Statistisches Amt des Kantons Zürich	Wähleranteil SP bei Nationalratswahlen. Hinweis: 2015 werden alle Auslandschweizer zur Stadt Zürich gezählt.
Ø steuerbares Einkommen natürliche Pers. [Fr.]	Franken	Statistisches Amt des Kantons Zürich	Durchschnittliches steuerbares Einkommen ordentlich besteuerter natürlichen Personen (exkl. Quellensteuerpflichtige und ausserhalb des Kt. wohnhafte). Hinweis: Sehr hohe Beträge einiger weniger Steuerpflichtigen wirken sich entsprechend stark auf den Mittelwert aus. Bis 1998: Reineinkommen.
Steuerkraft berichtigt pro Kopf [Fr.]	Franken pro Einwohner	Statistisches Amt des Kantons Zürich	Berichtigte Steuerkraft pro Einwohner (Steuerkraft nach Finanzausgleich pro Einwohner).
Nettoaufwand Umweltschutz und Raumordnung [Fr./Einw.]	Franken pro Einwohner	Statistisches Amt des Kantons Zürich, Gemeindefinanzstatistik (GEFIS)	Bruttoaufwand - Bruttoertrag (der Laufenden Rechnung). Aussage: Durch Steuern zu finanzierender Anteil des betreffenden Hauptaufgabenbereichs. Grundlage ist die Konsolidierte Gemeinde. Der Haushalt der Schulgemeinde wird dabei anteilsmässig in den Haushalt der poltischen Gemeinden umgelegt. Umstellung des Rechnungsmodells von HRM1 auf HRM2 ab Rechnungsjahr 2019.
Wasserverbrauch pro Tag und Kopf [Liter]	Liter pro Tag und Einwohner	Amt für Abfall, Wasser, Enerige und Luft (AWEL), Statistisches Amt des Kantons Zürich, Schweizerischer Verein des Gas- und Wasserfaches	Mittlerer Tagesverbrauch: Die Pro-Kopf-Werte müssen unter Vorbehalt betrachtet werden, da sie auch den Verbrauch von Industrie und Gewerbe mit einschliessen und deshalb nicht als haushaltstypisch angesehen werden können.
Bevölkerung [Pers.]	Personen	Statistisches Amt des Kantons Zürich, Kantonale Bevölkerungserhebung	Einwohnerbestand Ende Jahr nach zivilrechtlichem Wohnsitz (ab 2010 inkl. vorläufig Aufgenommene, die seit mehr als einem Jahr in der Gemeinde leben, aber ohne Wochenaufenthalter und Asylbewerber).
Altpapiermengen [Tonnen] / [Tonnen / Person]	Tonnen	Amt für Abfall, Wasser, Energie und Luft (AWEL)	Altpapiermengen aus kommunaler Sammlung von Industrie und Privathaushalten. Wo keine Angabe vorhanden sind, wird Papier zusammen mit Karton gesammelt.
Verpackungsglasmengen [Tonnen] / [Tonnen / Person]	Tonnen	Amt für Abfall, Wasser, Energie und Luft (AWEL)	Verpackungsglasmengen aus kommunaler Sammlung von Industrie und Privathaushalten.
Schül. Sekundarstufe II [Pers. pro 1000 Einw.]	Personen pro Tausend Einwohner	Bildungsstatistik Kanton Zürich	Die Sekundarstufe II umfasst die Berufsbildung und bei Mittelschulen die Normschuljahre > 9. Gemessen an der zivilrechtlichen Bevölkerung Ende Vorjahr
Bevölkerung: Durchschnittsalter [Jahre]	Jahre	Statistisches Amt des Kantons Zürich, Kantonale Bevölkerungserhebung	Durchschnittsalter Ende Jahr (zivilrechtlicher Wohnsitz; ab 2010 inkl. vorläufig Aufgenommene, die seit mehr als einem Jahr in der Gemeinde leben, aber ohne Wochenaufenthalter und Asylbewerber).
NRW Wähleranteil SVP [%]	Prozent	Statistisches Amt des Kantons Zürich	Wähleranteil SVP bei Nationalratswahlen. Hinweis: 2015 werden alle Auslandschweizer zur Stadt Zürich gezählt.
KRW Wähleranteil SVP [%]	Prozent	Statistisches Amt des Kantons Zürich	Wähleranteil SVP bei Kantonsratswahlen
Kehrichtmengen [Tonnen] / [Tonnen / Person]	Tonnen	Amt für Abfall, Wasser, Energie und Luft (AWEL)	Kehrichtmengen aus kommunaler Sammlung von Haushalten, ab 2004 mit Betriebsabfällen (Anteil Siedlungsabfall, ohne Produktionsabfälle). Nicht enthalten sind Direktanlieferungen von Hauhshalten und Betrieben an die KVA und private Entsorgungslösungen, die nicht über den kommunalen Weg laufen.

[1]:
```
# IMPORT PACKAGES

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from pandas_profiling import ProfileReport
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import colors as mcolors
```

[2]:
```
# CSV PATH - SETTTINGS

CSV_PATH = "/Users/simonsahli-studio/Documents/_a_Study/2022 MSc DBU/
 ↪2_Veranstaltungen/04 Data Mining/studienarbeit/code/"

#CSV_PATH = "/Users/simonsahli/Documents/_a_Study/2022 MSc DBU/2_Veranstaltungen/
 ↪04 Data Mining/studienarbeit/code/"
```

[3]:
```
# DATA IMPORT

## CSV: "zurich_gemeinden_alldata"

### Rows: 61387

### Columns: 11
### Columns: (['BFS_NR', 'GEBIET_NAME', 'THEMA_NAME', 'SET_NAME', 'SUBSET_NAME',
### 'INDIKATOR_ID', 'INDIKATOR_NAME', 'INDIKATOR_JAHR', 'INDIKATOR_VALUE',
### 'EINHEIT_KURZ', 'EINHEIT_LANG'],

zuerich_data = pd.read_csv(CSV_PATH + "zurich_gemeinden_alldata.csv")
zuerich_data.head()
```

[3]:

	BFS_NR	GEBIET_NAME	THEMA_NAME	SET_NAME	SUBSET_NAME	INDIKATOR_ID	INDIKATOR_NAME	INDIKATOR_JAHR	INDIKATOR_VALUE	EINHEIT_KURZ
0	1	Aeugst a.A.	Bevölkerung und Soziales	Bevölkerungsbestand und Haushalte	Bevölkerungsbestand	133				
1	1	Aeugst a.A.	Bevölkerung und Soziales	Bevölkerungsbestand und Haushalte	Bevölkerungsbestand	133				
2	1	Aeugst a.A.	Bevölkerung und Soziales	Bevölkerungsbestand und Haushalte	Bevölkerungsbestand	133				
3	1	Aeugst a.A.	Bevölkerung und Soziales	Bevölkerungsbestand und Haushalte	Bevölkerungsbestand	133				
4	1	Aeugst a.A.	Bevölkerung und Soziales	Bevölkerungsbestand und Haushalte	Bevölkerungsbestand	133				

```
0  Bevölkerung [Pers.]            1962            710.0        Pers.
1  Bevölkerung [Pers.]            1963            720.0        Pers.
2  Bevölkerung [Pers.]            1964            705.0        Pers.
3  Bevölkerung [Pers.]            1965            695.0        Pers.
4  Bevölkerung [Pers.]            1966            745.0        Pers.

  EINHEIT_LANG
0     Personen
1     Personen
2     Personen
3     Personen
4     Personen
```

[4]:
```
# CREATING ProfileReport

## ZURICH ALLDATA - REPORT:

""" plt.rcParams['font.family'] = 'serif'
plt.rcParams['font.serif'] = ['Times New Roman'] + plt.rcParams['font.serif']
profile = ProfileReport(zuerich_data, title="Pandas Profiling Report")
profile.to_file("zuerich_alldata_report.html")
"""

## ZURICH PIVOT DATA - TBD:
```

[4]:
```
' plt.rcParams[\'font.family\'] = \'serif\'\nplt.rcParams[\'font.serif\'] =
[\'Times New Roman\'] + plt.rcParams[\'font.serif\']\nprofile =
ProfileReport(zuerich_data, title="Pandas Profiling
Report")\nprofile.to_file("zuerich_alldata_report.html")\n'
```

[5]:
```
# FIRST DATA ANALYSIS (TYPE, FORM, TIME, NAN)
```

[6]:
```
zuerich_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 61387 entries, 0 to 61386
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   BFS_NR           61387 non-null  int64
 1   GEBIET_NAME      61387 non-null  object
 2   THEMA_NAME       61387 non-null  object
 3   SET_NAME         61387 non-null  object
 4   SUBSET_NAME      61387 non-null  object
 5   INDIKATOR_ID     61387 non-null  int64
 6   INDIKATOR_NAME   61387 non-null  object
 7   INDIKATOR_JAHR   61387 non-null  int64
```

```
 8   INDIKATOR_VALUE  61098 non-null  float64
 9   EINHEIT_KURZ     61387 non-null  object
 10  EINHEIT_LANG     61387 non-null  object
dtypes: float64(1), int64(3), object(7)
memory usage: 5.2+ MB
```

[7]:
```
zuerich_data.columns
```

[7]:
```
Index(['BFS_NR', 'GEBIET_NAME', 'THEMA_NAME', 'SET_NAME', 'SUBSET_NAME',
       'INDIKATOR_ID', 'INDIKATOR_NAME', 'INDIKATOR_JAHR', 'INDIKATOR_VALUE',
       'EINHEIT_KURZ', 'EINHEIT_LANG'],
      dtype='object')
```

[8]:
```
zuerich_data["INDIKATOR_NAME"].value_counts()
```

[8]:
```
Bevölkerung [Pers.]                                    10723
Ø steuerbares Einkommen natürliche Pers. [Fr.]          5767
Nettoaufwand Umweltschutz und Raumordnung [Fr./Einw.]   5604
Steuerkraft berichtigt pro Kopf [Fr.]                   5597
Arbeitslose [Anz.]                                      5446
Kehrichtmengen [Tonnen]                                 4503
Altpapiermengen [Tonnen]                                4503
Verpackungsglasmengen [Tonnen]                          4503
Schül. Sekundarstufe II [Pers. pro 1000 Einw.]          3916
Wasserverbrauch pro Tag und Kopf [Liter]                2556
Bevölkerung: Durchschnittsalter [Jahre]                 2396
NRW Wähleranteil SVP [%]                                1355
NRW Wähleranteil SP [%]                                 1355
KRW Wähleranteil SVP [%]                                 829
KRW Wähleranteil SP [%]                                  829
KRW Wähleranteil GLP [%]                                 829
NRW Wähleranteil GLP [%]                                 676
Name: INDIKATOR_NAME, dtype: int64
```

[9]:
```
print(zuerich_data['GEBIET_NAME'].value_counts(), zuerich_data['INDIKATOR_JAHR'].
↪value_counts())
```

```
Aeugst a.A.               355
Zumikon                   355
Hombrechtikon             355
Küsnacht                  355
Männedorf                 355
                         ...
Wädenswil                 272
Illnau-Effretikon         265
Bauma (bis 2014)          260
Sternenberg (bis 2014)    260
Andelfingen               259
```

```
Name: GEBIET_NAME, Length: 177, dtype: int64 2011    2951
2015    2932
2007    2774
2019    2763
2014    1931
        ...
1985     177
1986     177
1987     177
1988     177
1962     177
Name: INDIKATOR_JAHR, Length: 62, dtype: int64
```

[10]:
```
plt.figure(figsize=[5,4])
sns.histplot(data=zuerich_data, x='INDIKATOR_JAHR', bins=30, kde=True)
plt.title("Abbildung 1: Verteilung aller Beobachtungen über die Zeit")
plt.xlabel("Beobachtungszeitraum")
plt.show()
```

[11]:
```
zuerich_data[zuerich_data["INDIKATOR_JAHR"] <= 1990]["INDIKATOR_NAME"].unique()
```

[11]:
```
array(['Bevölkerung [Pers.]',
       'Nettoaufwand Umweltschutz und Raumordnung [Fr./Einw.]',
       'Ø steuerbares Einkommen natürliche Pers. [Fr.]',
       'Steuerkraft berichtigt pro Kopf [Fr.]'], dtype=object)
```

[12]:
```
print(zuerich_data[zuerich_data["INDIKATOR_NAME"] == "Kehrichtmengen␣
↪[Tonnen]"]["INDIKATOR_JAHR"].unique(),␣
↪len(zuerich_data[zuerich_data["INDIKATOR_NAME"] == "Kehrichtmengen␣
↪[Tonnen]"]["INDIKATOR_JAHR"].unique()))
```

```
[1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009
 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021] 26
```

[13]:
```
plt.figure(figsize=[5,4])
scatter = sns.scatterplot(data=zuerich_data, x="INDIKATOR_JAHR",␣
↪y="INDIKATOR_NAME", hue="INDIKATOR_NAME", palette='colorblind', alpha=0.5,␣
↪legend=False)
plt.title('Number of observations over time for each INDICATOR_NAME')
plt.ylabel('')
plt.xlabel("Beobachtungszeitraum")
plt.title("Abbildung 2: Vorhandene Daten je Datensatz über die Zeit")
plt.show()
```

[14]:
```
nan_rows = zuerich_data[zuerich_data["INDIKATOR_VALUE"].isnull()]
print(nan_rows[["GEBIET_NAME", "SUBSET_NAME", "INDIKATOR_JAHR"]], nan_rows.shape)
```

```
             GEBIET_NAME       SUBSET_NAME  INDIKATOR_JAHR
192          Aeugst a.A.          Verbrauch            2013
2424         Maschwanden  Arbeitslosigkeit            2016
2426         Maschwanden  Arbeitslosigkeit            2018
2427         Maschwanden  Arbeitslosigkeit            2019
2428         Maschwanden  Arbeitslosigkeit            2020
...                  ...               ...             ...
54770          Weiningen            Abfall            2010
55695          Stammheim          Verbrauch            2018
56742  Illnau-Effretikon          Verbrauch            2010
57300       Wiesendangen            Abfall            2009
57301       Wiesendangen            Abfall            2010

[289 rows x 3 columns] (289, 11)
```

[15]:
```
nan_counts_sum = nan_rows.groupby("GEBIET_NAME").size()
nan_counts_sum = nan_counts_sum.sort_values(ascending=False)
nan_counts_sum

plt.figure(figsize=[20,4])
nan_counts_sum.plot(kind="bar")
plt.ylabel('')
plt.xlabel('')
plt.title("Abbildung A: Fehlende Dateneinträge je Gemeinde")
plt.show()
```

[16]:
```
## How many communities in the canton of Zurich have at least one missing data␣
 ↪entry in the dataset?
nan_counts_sum.head
```

[16]:
```
<bound method NDFrame.head of GEBIET_NAME
Volken                 27
Humlikon (bis 2022)    18
Wasterkingen           14
Truttikon              14
Bachs                  12
```

```
                        ..
Höri                     1
Illnau-Effretikon        1
Kloten                   1
Aeugst a.A.              1
Lindau                   1
Length: 65, dtype: int64>
```

[17]:
```python
# 3D Plot of NaN-Values

pivot_table = nan_rows.groupby(["GEBIET_NAME", "INDIKATOR_NAME"]).size().
 ↪unstack(fill_value=0)
pivot_table = pivot_table.reindex(pivot_table.sum(axis=1).
 ↪sort_values(ascending=False).index).head(40)

## Preparing Colors
colors = list(mcolors.CSS4_COLORS.keys())  # Get a list of valid color names
color_dict = {gebiet: colors[i] for i, gebiet in enumerate(pivot_table.index)}

## Preparing Bars
num_bars = len(pivot_table.index)
gap = 0.5  # distance between bars
x_pos = np.arange(num_bars) * (1 + gap)
z_pos = np.zeros(num_bars)
dx = dy = np.ones(num_bars)

fig = plt.figure(figsize=(40,20))
ax = fig.add_subplot(111, projection='3d')

## Generating Bars
for i, indikator_name in enumerate(pivot_table.columns):
    y_pos = (np.ones(num_bars) * i) * (1 + gap)
    dz = pivot_table[indikator_name].values
    color = [color_dict[gebiet] for gebiet in pivot_table.index]
    ax.bar3d(x_pos, y_pos, z_pos, dx, dy, dz, color=color, alpha=0.5)

## X-Achsis
ax.set_xticks(x_pos)
ax.set_xticklabels(pivot_table.index, rotation=90, fontsize=8)

## Y-Achsis
ax.set_yticks(np.arange(len(pivot_table.columns)) * (1 + gap) + 0.8)
ax.set_yticklabels(pivot_table.columns, rotation=-20, fontsize=8)

## Label s
# ax.set_xlabel('GEBIET_NAME')
# ax.set_ylabel('INDIKATOR_NAME')
```

```
ax.set_title("Abbildung 3: 3D-Visualisierung der fehlenden Datenwerte aus 40␣
 ↪Gemeinden des Kantons Zürich")
ax.set_zlabel('Count of NaN values')

plt.show()
```

[18]:
```
volken = zuerich_data[zuerich_data["GEBIET_NAME"]=="Volken"]
volken[(volken["INDIKATOR_NAME"] == "Altpapiermengen [Tonnen]") |␣
 ↪(volken["INDIKATOR_NAME"] == "Arbeitslose [Anz.]")].head()
```

[18]:

	BFS_NR	GEBIET_NAME	THEMA_NAME	SET_NAME	SUBSET_NAME	INDIKATOR_ID	INDIKATOR_NAME	INDIKATOR_JAHR	INDIKATOR_VALUE	EINHEIT_KURZ	EINHEIT_LANG
11928	43	Volken	Arbeit und Unternehmen	Arbeitslosigkeit	Arbeitslosigkeit	107	Arbeitslose [Anz.]	1991	0.8	Anz.	Anzahl
11929	43	Volken	Arbeit und Unternehmen	Arbeitslosigkeit	Arbeitslosigkeit	107	Arbeitslose [Anz.]	1992	1.2	Anz.	Anzahl
11930	43	Volken	Arbeit und Unternehmen	Arbeitslosigkeit	Arbeitslosigkeit	107	Arbeitslose [Anz.]	1993	0.3	Anz.	Anzahl
11931	43	Volken	Arbeit und Unternehmen	Arbeitslosigkeit	Arbeitslosigkeit	107	Arbeitslose [Anz.]	1994	0.5	Anz.	Anzahl
11932	43	Volken	Arbeit und Unternehmen	Arbeitslosigkeit	Arbeitslosigkeit	107	Arbeitslose [Anz.]	1995	1.9	Anz.	Anzahl

[19]:
```
# DELETING NAN-VALUES

print(zuerich_data.shape) ## before deleting NaN-Values of "INDIKATOR_VALUE"
zuerich_data = zuerich_data.dropna(subset=['INDIKATOR_VALUE'])
print(zuerich_data.shape) ## after deleting NaN-Values of "INDIKATOR_VALUE"
```

```
(61387, 11)
(61098, 11)
```

[20]:
```
# TRANSFORMATION OF DATA FRAMES TO PIVOT DATA FRAMES FOR FURTHER EDA AND DATA␣
 ↪MODELING
```

```
zurich_pivot_data = zuerich_data.pivot_table(index=["GEBIET_NAME",␣
 ↪"INDIKATOR_JAHR"],
                                             columns="INDIKATOR_ID",
                                             values="INDIKATOR_VALUE",
                                             aggfunc="mean")
## aggfunc = mean irrelevant, as for each indicator value and each year and ID␣
 ↪only one observation

zurich_pivot_data.reset_index(inplace=True)
zurich_pivot_data.columns.name = None
print(zurich_pivot_data.head(), zurich_pivot_data.shape)
```

```
          GEBIET_NAME  INDIKATOR_JAHR  107  118  119  124  125    133  134  \
0  Adlikon (bis 2022)            1962  NaN  NaN  NaN  NaN  NaN  400.0  NaN
1  Adlikon (bis 2022)            1963  NaN  NaN  NaN  NaN  NaN  400.0  NaN
2  Adlikon (bis 2022)            1964  NaN  NaN  NaN  NaN  NaN  389.0  NaN
3  Adlikon (bis 2022)            1965  NaN  NaN  NaN  NaN  NaN  388.0  NaN
4  Adlikon (bis 2022)            1966  NaN  NaN  NaN  NaN  NaN  390.0  NaN

   135  138  274  282  427  438  442  465  469  809
0  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN
1  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN
2  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN
3  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN
4  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN  NaN   (10883, 19)
```

[21]:
```
## Cleaning up the naming of the pivot data frame

zurich_pivot_data = zurich_pivot_data.rename({
    133 : "population",
    809 : "avg_age",
    465 : "students_per_1000pop",
    107 : "unemployment",
    135 : "garbage",
    134 : "paper",
    138 : "glas",
    469 : "water",
    427 : "exp_environmental_protection_capita",
    282 : "taxable_income_natural",
    274 : "taxpower_adj_capita",
    125 : "nrw_svp",
    124 : "nrw_sp",
    442 : "nrw_glp",
    119 : "krw_svp",
    118 : "krw_sp",
    438 : "krw_glp",
}, axis=1)
```

[22]:
```
## Creating 4 new capita variables

zurich_pivot_data["garbage_capita"] = zurich_pivot_data["garbage"]/
 ↪zurich_pivot_data["population"]
zurich_pivot_data["paper_capita"] = zurich_pivot_data["paper"]/
 ↪zurich_pivot_data["population"]
zurich_pivot_data["glas_capita"] = zurich_pivot_data["glas"]/
 ↪zurich_pivot_data["population"]
zurich_pivot_data["unemployment_capita"] = zurich_pivot_data["unemployment"]/
 ↪zurich_pivot_data["population"]
```

[23]:
```
zurich_pivot_data.columns
```

[23]:
```
Index(['GEBIET_NAME', 'INDIKATOR_JAHR', 'unemployment', 'krw_sp', 'krw_svp',
       'nrw_sp', 'nrw_svp', 'population', 'paper', 'garbage', 'glas',
       'taxpower_adj_capita', 'taxable_income_natural',
       'exp_environmental_protection_capita', 'krw_glp', 'nrw_glp',
       'students_per_1000pop', 'water', 'avg_age', 'garbage_capita',
       'paper_capita', 'glas_capita', 'unemployment_capita'],
      dtype='object')
```

[24]:
```
columns_to_plot = ['unemployment_capita',
       'nrw_sp', 'nrw_svp','krw_sp', 'krw_svp', 'population',␣
 ↪'taxpower_adj_capita', 'taxable_income_natural',
       'students_per_1000pop', 'exp_environmental_protection_capita',
       'nrw_glp', 'water', 'avg_age', 'garbage_capita', 'paper_capita',
       'glas_capita']
```

[25]:
```
# HISTOGRAMS OF RELEVANT DATA SETS WITHIN THE DATA FRAME

columns_to_plot = ['unemployment_capita',
       'nrw_sp', 'nrw_svp','krw_sp', 'krw_svp', 'population',␣
 ↪'taxpower_adj_capita', 'taxable_income_natural',
       'students_per_1000pop', 'exp_environmental_protection_capita',
       'nrw_glp', 'water', 'avg_age', 'garbage_capita', 'paper_capita',
       'glas_capita']

x_labels = {    'unemployment_capita' : "Personen",
                'nrw_sp' : "%",
                'nrw_svp' : "%",
                'krw_sp' : "%",
                'krw_svp' : "%",
                'population' : "Personen",
                'taxpower_adj_capita' : "CHF",
                'taxable_income_natural' : "CHF",
                'students_per_1000pop' : "Personen",
                'exp_environmental_protection_capita' : "CHF",
```

```
                'nrw_glp' : "%",
                'water' : "Liter",
                'avg_age' : "Jahre",
                'garbage_capita' : "Tonnen",
                'paper_capita' : "Tonnen",
                'glas_capita' : "Tonnen"}

titles = {      'unemployment_capita' : "Personen [capita]",
                'nrw_sp' : "SP NRW-Wähleranteil",
                'nrw_svp' : "SVP NRW-Wähleranteil",
                'krw_sp' : "SP KRW-Wähleranteil",
                'krw_svp' : "SVP NRW-Wähleranteil",
                'population' : "Bevölkerung",
                'taxpower_adj_capita' : "Steuerkraft bereinigt [capita]",
                'taxable_income_natural' : "Ø-Einkommen nat. Person",
                'students_per_1000pop' : "Anzahl Studierende auf 1000 Personen",
                'exp_environmental_protection_capita' : "Nettoaufwand␣
 ↪Umwerltschutz [capita]",
                'nrw_glp' : "GLP NRW-Wähleranteil",
                'water' : "Wasserverbrauch [tag/capita]",
                'avg_age' : "Durchchnittsalter Bevölkerung",
                'garbage_capita' : "Kehrichtmenge [capita]",
                'paper_capita' : "Altpapier [capita]",
                'glas_capita' : "Altglas [capita]"}

# Number of rows and columns in subplot grid
n_rows = 4
n_cols = 4

fig, axes = plt.subplots(n_rows, n_cols, figsize=(15, 15))

fig.suptitle("Abbildung 4: Überblick über die Verteilungen verschiedener␣
 ↪Datensatz-Variablen", fontsize=10)
fig.tight_layout(pad=5.0)

for i, column in enumerate(columns_to_plot):
    # Compute the row index and column index
    row_idx = i // n_cols
    col_idx = i % n_cols

    # Get the current axis
    ax = axes[row_idx, col_idx]

    # Plot histogram on the current axis
    sns.histplot(zurich_pivot_data[column], kde=False, ax=ax)

    # Set title
```

```
    ax.set_title(titles[column])
    ax.set_xlabel(x_labels[column])

    # Calculate statistics
    avg = zurich_pivot_data[column].mean()
    min_value = zurich_pivot_data[column].min()
    max_value = zurich_pivot_data[column].max()
    std_dev = zurich_pivot_data[column].std()

    # Display statistics on the plot
    stats_text = f'Avg: {avg:.2f}\nMin: {min_value:.2f}\nMax: {max_value:.
↪2f}\nSD: {std_dev:.2f}'
    ax.text(0.70, 0.95, stats_text, transform=ax.transAxes,␣
↪verticalalignment='top', fontsize=6)

plt.show()
```

[26]:
```
# IDENTIFYING AND TREATING OUTLIERS BY USE OF THE 1.5*IQR RULE

zurich_pivot_data.shape
```

[26]:
```
(10883, 23)
```

[27]:
```
## Outliers: Unemplyoment Capita

Q1 = zurich_pivot_data['unemployment_capita'].quantile(0.25)
Q3 = zurich_pivot_data['unemployment_capita'].quantile(0.75)
IQR = Q3 - Q1

outliers = zurich_pivot_data[(zurich_pivot_data['unemployment_capita'] < (Q1 - 1.
↪5 * IQR)) | (zurich_pivot_data['unemployment_capita'] > (Q3 + 1.5 * IQR))]
outliers.query("GEBIET_NAME == 'Opfikon' | GEBIET_NAME == 'Dietikon'")␣
↪[["INDIKATOR_JAHR", "GEBIET_NAME", "unemployment_capita", "unemployment"]].
↪sort_values(by="unemployment_capita", ascending=False).head(15)
## Notable: Opfikon, Dietikon
```

[27]:

	INDIKATOR_JAHR	GEBIET_NAME	unemployment_capita	unemployment
6793	1997	Opfikon	0.048811	539.7
1636	1997	Dietikon	0.045666	934.5
6800	2004	Opfikon	0.045189	573.0
6799	2003	Opfikon	0.044608	565.1
6801	2005	Opfikon	0.042100	538.0
1633	1994	Dietikon	0.041397	876.2

```
1635            1996    Dietikon            0.041200        854.2
6794            1998     Opfikon            0.040331        452.8
1632            1993    Dietikon            0.038410        811.8
6790            1994     Opfikon            0.037982        436.3
1644            2005    Dietikon            0.036283        802.0
1634            1995    Dietikon            0.035428        743.5
1643            2004    Dietikon            0.034827        760.0
1642            2003    Dietikon            0.034796        756.7
6798            2002     Opfikon            0.033875        417.2
```

[28]:
```
## Outliers: Environmental protection expenditure per capita

Q1 = zurich_pivot_data['exp_environmental_protection_capita'].quantile(0.25)
Q3 = zurich_pivot_data['exp_environmental_protection_capita'].quantile(0.75)
IQR = Q3 - Q1

outliers =␣
↪zurich_pivot_data[(zurich_pivot_data['exp_environmental_protection_capita'] <␣
↪(Q1 - 1.5 * IQR)) | (zurich_pivot_data['exp_environmental_protection_capita']␣
↪> (Q3 + 1.5 * IQR))]
outliers[["GEBIET_NAME", "INDIKATOR_JAHR",␣
↪"exp_environmental_protection_capita"]].
↪sort_values(by="exp_environmental_protection_capita")
## Notable: Sternenberg (bis 2014)
```

[28]:
```
                GEBIET_NAME  INDIKATOR_JAHR  \
6238             Oberembrach            1995
2744              Fehraltorf            1995
9092            Uetikon a.S.            2016
6236             Oberembrach            1993
5557                    Maur            1996
...                      ...             ...
32        Adlikon (bis 2022)            1994
4465               Hüntwangen            1999
2492         Ellikon a.d.Th.             1991
5126                Küsnacht            1999
8665  Sternenberg (bis 2014)            2014

      exp_environmental_protection_capita
6238                                -397.0
2744                                -327.0
9092                                -274.0
6236                                -227.0
5557                                -194.0
...                                    ...
32                                  467.0
4465                                496.0
```

```
2492                                523.0
5126                                555.0
8665                               2387.0

[274 rows x 3 columns]
```

[29]:
```
### Checking and Cleaning of: Sternenberg (until 2014)

zurich_pivot_data.query("GEBIET_NAME == 'Sternenberg (bis␣
↪2014)'")[["GEBIET_NAME", "INDIKATOR_JAHR",␣
↪"exp_environmental_protection_capita" ]].dropna()
```

[29]:
```
                 GEBIET_NAME  INDIKATOR_JAHR  \
8641  Sternenberg (bis 2014)            1990
8642  Sternenberg (bis 2014)            1991
8643  Sternenberg (bis 2014)            1992
8644  Sternenberg (bis 2014)            1993
8645  Sternenberg (bis 2014)            1994
8646  Sternenberg (bis 2014)            1995
8647  Sternenberg (bis 2014)            1996
8648  Sternenberg (bis 2014)            1997
8649  Sternenberg (bis 2014)            1998
8650  Sternenberg (bis 2014)            1999
8651  Sternenberg (bis 2014)            2000
8652  Sternenberg (bis 2014)            2001
8653  Sternenberg (bis 2014)            2002
8654  Sternenberg (bis 2014)            2003
8655  Sternenberg (bis 2014)            2004
8656  Sternenberg (bis 2014)            2005
8657  Sternenberg (bis 2014)            2006
8658  Sternenberg (bis 2014)            2007
8659  Sternenberg (bis 2014)            2008
8660  Sternenberg (bis 2014)            2009
8661  Sternenberg (bis 2014)            2010
8662  Sternenberg (bis 2014)            2011
8663  Sternenberg (bis 2014)            2012
8664  Sternenberg (bis 2014)            2013
8665  Sternenberg (bis 2014)            2014

      exp_environmental_protection_capita
8641                                103.0
8642                                201.0
8643                                177.0
8644                                167.0
8645                                 73.0
8646                                 72.0
8647                                 89.0
```

```
8648                                81.0
8649                                97.0
8650                               115.0
8651                                90.0
8652                               105.0
8653                                79.0
8654                               150.0
8655                               101.0
8656                               104.0
8657                               110.0
8658                               120.0
8659                               145.0
8660                               128.0
8661                               163.0
8662                               159.0
8663                               135.0
8664                               160.0
8665                              2387.0
```

[30]:
```
condition = (zurich_pivot_data['GEBIET_NAME'] != 'Sternenberg (bis 2014)') |␣
 ↪(zurich_pivot_data['INDIKATOR_JAHR'] != 2014)
zurich_pivot_data = zurich_pivot_data[condition]
zurich_pivot_data.shape # 10883 - 1 = 10882
```

[30]:
```
(10882, 23)
```

[31]:
```
## Outliers: Water usage

Q1 = zurich_pivot_data['water'].quantile(0.25)
Q3 = zurich_pivot_data['water'].quantile(0.75)
IQR = Q3 - Q1
outliers = zurich_pivot_data[(zurich_pivot_data['water'] < (Q1 - 1.5 * IQR)) |␣
 ↪(zurich_pivot_data['water'] > (Q3 + 1.5 * IQR))]
outliers[["GEBIET_NAME", "INDIKATOR_JAHR", "water"]].sort_values(by="water")
## Notable: Hüttikon, 2006
```

[31]:
```
          GEBIET_NAME  INDIKATOR_JAHR   water
7367          Rheinau            2013   399.0
1957         Dällikon            2008   399.0
846             Benken            2013   400.0
1955         Dällikon            2006   400.0
2509  Ellikon a.d.Th.            2008   401.0
...               ...             ...     ...
915         Berg a.I.            2020   577.0
914         Berg a.I.            2019   581.0
2514  Ellikon a.d.Th.            2013   671.0
4592         Hüttikon            2007   811.0
```

```
4591         Hüttikon            2006  1036.0

[90 rows x 3 columns]
```

[32]:
```
wasserverbrauch = zurich_pivot_data.groupby(["INDIKATOR_JAHR"])["water"].mean()
wasserverbrauch.dropna().sort_values()
```

[32]:
```
INDIKATOR_JAHR
2019    242.192547
2016    245.458824
2020    248.105590
2017    248.428571
2014    250.272189
2012    252.928994
2018    255.216049
2013    258.747059
2010    259.325000
2015    260.181287
2011    260.643275
2009    268.710059
2008    276.094118
2007    280.529412
2006    295.917647
Name: water, dtype: float64
```

[33]:
```
### Checking and Cleaning of: Hüttikon

hüttikon = zurich_pivot_data.query("GEBIET_NAME== 'Hüttikon'")[["GEBIET_NAME",␣
 ↪"population", "INDIKATOR_JAHR", "water" ]]
hüttikon["water_sum"] = hüttikon["population"]*hüttikon["water"]
hüttikon.dropna()
```

[33]:

	GEBIET_NAME	population	INDIKATOR_JAHR	water	water_sum
4591	Hüttikon	560.0	2006	1036.0	580160.0
4592	Hüttikon	627.0	2007	811.0	508497.0
4593	Hüttikon	666.0	2008	492.0	327672.0
4594	Hüttikon	654.0	2009	395.0	258330.0
4595	Hüttikon	665.0	2010	408.0	271320.0
4596	Hüttikon	664.0	2011	409.0	271576.0
4597	Hüttikon	668.0	2012	274.0	183032.0
4598	Hüttikon	679.0	2013	249.0	169071.0
4599	Hüttikon	703.0	2014	330.0	231990.0
4600	Hüttikon	879.0	2015	251.0	220629.0
4601	Hüttikon	916.0	2016	229.0	209764.0
4602	Hüttikon	925.0	2017	213.0	197025.0
4603	Hüttikon	922.0	2018	258.0	237876.0
4604	Hüttikon	949.0	2019	260.0	246740.0

```
4605    Hüttikon       947.0            2020    257.0    243379.0
```

[34]:
```
zurich_pivot_data.query("GEBIET_NAME== 'Hüttikon' & INDIKATOR_JAHR == 2006")
condition = (zurich_pivot_data['GEBIET_NAME'] != 'Hüttikon') |␣
↪(zurich_pivot_data['INDIKATOR_JAHR'] != 2006)
zurich_pivot_data = zurich_pivot_data[condition]
zurich_pivot_data.shape # 10882 - 1 = 10881
```

[34]:
```
(10881, 23)
```

[35]:
```
## Outliers: Garbage_capita

Q1 = zurich_pivot_data['garbage_capita'].quantile(0.25)
Q3 = zurich_pivot_data['garbage_capita'].quantile(0.75)
IQR = Q3 - Q1

outliers = zurich_pivot_data[(zurich_pivot_data['garbage_capita'] < (Q1 - 1.5 *␣
↪IQR)) | (zurich_pivot_data['garbage_capita'] > (Q3 + 1.5 * IQR))]
outliers[["GEBIET_NAME", "INDIKATOR_JAHR", "garbage_capita"]].
↪sort_values(by="garbage_capita")
## Notable: All observations below threshold
```

[35]:
```
                  GEBIET_NAME  INDIKATOR_JAHR  garbage_capita
4231                   Horgen            2006        0.017333
229                Aeugst a.A.           2006        0.021898
2093                 Dättlikon           2020        0.052566
2094                 Dättlikon           2021        0.052963
5498               Maschwanden           1999        0.069680
...                        ...            ...             ...
9636               Wallisellen           2007        0.377554
9695    Waltalingen (bis 2018)           2004        0.379009
6487  Oberstammheim (bis 2018)           1996        0.383142
9638               Wallisellen           2009        0.440336
9637               Wallisellen           2008        0.493100

[81 rows x 3 columns]
```

[36]:
```
zurich_pivot_data.groupby(["INDIKATOR_JAHR"])["garbage_capita"].mean().dropna()
```

[36]:
```
INDIKATOR_JAHR
1996    0.169019
1997    0.152814
1998    0.151424
1999    0.148903
2000    0.150988
2001    0.150409
2002    0.151222
```

```
2003    0.152664
2004    0.183001
2005    0.183558
2006    0.185684
2007    0.187693
2008    0.189969
2009    0.187233
2010    0.187568
2011    0.185731
2012    0.187091
2013    0.185183
2014    0.184256
2015    0.181930
2016    0.180155
2017    0.179047
2018    0.179871
2019    0.175990
2020    0.177645
2021    0.175216
Name: garbage_capita, dtype: float64
```

[37]:
```
zurich_pivot_data[zurich_pivot_data['garbage_capita'] < 0.08] # All observations
↪below threshold
```

[37]:
```
            GEBIET_NAME  INDIKATOR_JAHR  unemployment  krw_sp  krw_svp  \
229         Aeugst a.A.            2006          23.0     NaN      NaN
1949           Dällikon            2000          34.3     NaN      NaN
2093          Dättlikon            2020           6.8     NaN      NaN
2094          Dättlikon            2021           NaN     NaN      NaN
4231             Horgen            2006           NaN     NaN      NaN
5073  Kyburg (bis 2015)            2000           1.5     NaN      NaN
5497        Maschwanden            1998           3.2     NaN      NaN
5498        Maschwanden            1999           1.7     NaN      NaN
5499        Maschwanden            2000           1.3     NaN      NaN

      nrw_sp  nrw_svp  population   paper  garbage  ...  \
229      NaN      NaN      1644.0   125.0     36.0  ...
1949     NaN      NaN      3228.0   209.0    246.0  ...
2093     NaN      NaN       799.0    32.0     42.0  ...
2094     NaN      NaN       793.0    31.0     42.0  ...
4231     NaN      NaN     19846.0  1543.0    344.0  ...
5073     NaN      NaN       391.0    21.0     31.0  ...
5497     NaN      NaN       501.0    30.0     35.0  ...
5498    28.2     50.4       531.0    34.0     37.0  ...
5499     NaN      NaN       574.0    36.0     44.0  ...

      exp_environmental_protection_capita  krw_glp  nrw_glp  \
```

```
229                                  81.0      NaN      NaN
1949                                 65.0      NaN      NaN
2093                                 72.0      NaN      NaN
2094                                 83.0      NaN      NaN
4231                                 56.0      NaN      NaN
5073                                122.0      NaN      NaN
5497                                 77.0      NaN      NaN
5498                                 50.0      NaN      NaN
5499                                 39.0      NaN      NaN

      students_per_1000pop  water  avg_age  garbage_capita  paper_capita  \
229                   47.9  248.0      NaN        0.021898      0.076034
1949                  34.7    NaN      NaN        0.076208      0.064746
2093                  47.6  170.0     42.1        0.052566      0.040050
2094                  57.6    NaN     43.1        0.052963      0.039092
4231                   NaN    NaN      NaN        0.017333      0.077749
5073                  50.9    NaN      NaN        0.079284      0.053708
5497                   NaN    NaN      NaN        0.069860      0.059880
5498                  14.0    NaN      NaN        0.069680      0.064030
5499                  28.2    NaN      NaN        0.076655      0.062718

      glas_capita  unemployment_capita
229      0.032847             0.013990
1949     0.025403             0.010626
2093     0.043805             0.008511
2094     0.041614                  NaN
4231     0.032349                  NaN
5073     0.030691             0.003836
5497     0.005988             0.006387
5498     0.007533             0.003202
5499     0.005226             0.002265

[9 rows x 23 columns]
```

[38]:
```
zurich_pivot_data = zurich_pivot_data[~(zurich_pivot_data['garbage_capita'] < 0.
↪08)]
zurich_pivot_data.shape # 10881 - 9 = 10872
```

[38]:
```
(10872, 23)
```

[39]:
```
## Outliers: Paper_capita
Q1 = zurich_pivot_data['paper_capita'].quantile(0.25)
Q3 = zurich_pivot_data['paper_capita'].quantile(0.75)
IQR = Q3 - Q1

outliers = zurich_pivot_data[(zurich_pivot_data['paper_capita'] < (Q1 - 1.5 *␣
↪IQR)) | (zurich_pivot_data['paper_capita'] > (Q3 + 1.5 * IQR))]
```

```
outliers[["GEBIET_NAME", "INDIKATOR_JAHR", "paper_capita"]].
 ↪sort_values(by="paper_capita")
## Notable: All observations below and above threshold
```

[39]:

	GEBIET_NAME	INDIKATOR_JAHR	paper_capita
9461	Volken	2018	0.000000
6261	Oberembrach	2018	0.000000
6138	Niederweningen	2019	0.000000
5209	Langnau a.A.	2020	0.000000
5208	Langnau a.A.	2019	0.000000
...	...	...	...
410	Andelfingen	2001	0.171823
472	Andelfingen (bis 2022)	2001	0.229429
9705	Waltalingen (bis 2018)	2014	0.235556
9448	Volken	2005	0.309524
4545	Hütten (bis 2018)	2017	1.075978

[75 rows x 3 columns]

[40]:
```
zurich_pivot_data.groupby(["INDIKATOR_JAHR"])["paper_capita"].mean().dropna()
```

[40]:

INDIKATOR_JAHR	
1996	0.067103
1997	0.067597
1998	0.070460
1999	0.074313
2000	0.076585
2001	0.077422
2002	0.073296
2003	0.069651
2004	0.071956
2005	0.073264
2006	0.072922
2007	0.071928
2008	0.071153
2009	0.065133
2010	0.062621
2011	0.063367
2012	0.059066
2013	0.055319
2014	0.054424
2015	0.048531
2016	0.046185
2017	0.049282
2018	0.037955
2019	0.036917
2020	0.032259

```
2021    0.027889
Name: paper_capita, dtype: float64
```

[41]:
```
zurich_pivot_data = zurich_pivot_data[~(zurich_pivot_data['paper_capita'] < 0.
↪01)]
condition = (zurich_pivot_data['GEBIET_NAME'] != 'Hütten (bis 2018)') |␣
↪(zurich_pivot_data['INDIKATOR_JAHR'] != 2017)
zurich_pivot_data = zurich_pivot_data[condition]
zurich_pivot_data.shape #10872 - 50 = 10822
```

[41]:
```
(10822, 23)
```

[42]:
```
## Outliers: Glas_capita
Q1 = zurich_pivot_data['glas_capita'].quantile(0.25)
Q3 = zurich_pivot_data['glas_capita'].quantile(0.75)
IQR = Q3 - Q1

outliers = zurich_pivot_data[(zurich_pivot_data['glas_capita'] < (Q1 - 1.5 *␣
↪IQR)) | (zurich_pivot_data['glas_capita'] > (Q3 + 1.5 * IQR))]
outliers[["GEBIET_NAME", "INDIKATOR_JAHR", "glas_capita"]].
↪sort_values(by="glas_capita")
## Notable: All observations below and above threshold
```

[42]:

	GEBIET_NAME	INDIKATOR_JAHR	glas_capita
6202	Nürensdorf	2021	0.000000
6201	Nürensdorf	2020	0.000000
6140	Niederweningen	2021	0.000000
9198	Unterengstringen	1998	0.002211
4103	Hofstetten (bis 2017)	1996	0.002326
...	...	...	...
530	Bachenbülach	1998	0.105348
531	Bachenbülach	1999	0.112476
532	Bachenbülach	2000	0.117685
10001	Weisslingen	2005	0.118430
1845	Dorf	2020	1.263158

[152 rows x 3 columns]

[43]:
```
zurich_pivot_data.query("glas_capita < 0.005")
zurich_pivot_data = zurich_pivot_data[~(zurich_pivot_data['glas_capita'] < 0.
↪005)]
zurich_pivot_data.query("GEBIET_NAME== 'Dorf' & INDIKATOR_JAHR == 2020")
condition = (zurich_pivot_data['GEBIET_NAME'] != 'Dorf') |␣
↪(zurich_pivot_data['INDIKATOR_JAHR'] != 2020)
zurich_pivot_data = zurich_pivot_data[condition]
zurich_pivot_data.shape # 10822 - 7 = 10815
```

[43]:
```
(10815, 23)
```

[44]:
```
# SAVING A BACKUP CSV FILE WITH TRANSFORMED AND CLEANED PIVOT DATA

## Uncomment to run:
## zurich_pivot_data.to_csv(CSV_PATH + "zurich_pivot_data.csv", index=False)
```

[45]:
```
# PREPARING DATA FOR CLUSTERING

## Correlation analysis

zurich_pivot_data_corr = zurich_pivot_data[['unemployment_capita', 'krw_sp',␣
↪'krw_svp',
       'nrw_sp', 'nrw_svp', 'population', 'paper_capita', 'garbage_capita',␣
↪'glas_capita',
       'taxpower_adj_capita', 'taxable_income_natural', 'students_per_1000pop',
       'exp_environmental_protection_capita', 'krw_glp', 'nrw_glp', 'water',
       'avg_age']].dropna().corr()

sns.heatmap(zurich_pivot_data_corr, annot=True, annot_kws={"size": 5},␣
↪cmap='binary')
plt.title('Abbildung B: Korrelation zwischen verschiedenen Variablen innerhalb␣
↪des DataFrames')
plt.xticks(fontsize=6)
plt.yticks(fontsize=6)
plt.show()
```

[46]:
```
## Annotated data frame, i.e. with region and year:
cluster_data_annot = zurich_pivot_data[['GEBIET_NAME', 'INDIKATOR_JAHR',␣
↪'unemployment_capita',
       'nrw_svp', 'taxable_income_natural', 'students_per_1000pop',␣
↪'exp_environmental_protection_capita',
       'garbage_capita']]
```

[47]:
```
cluster_data_annot.shape
```

[47]:
```
(10815, 8)
```

[48]:
```
cluster_data_annot.isnull().sum()
```

[48]:
```
GEBIET_NAME                0
INDIKATOR_JAHR             0
unemployment_capita     5569
```

```
nrw_svp                                9468
taxable_income_natural                 5116
students_per_1000pop                   6961
exp_environmental_protection_capita    5279
garbage_capita                         6382
dtype: int64
```

[49]:
```
cluster_data_annot = cluster_data_annot.dropna(subset=['garbage_capita'])
cluster_data_annot = cluster_data_annot.reset_index(drop=True)
cluster_data_annot.shape
```

[49]:
```
(4433, 8)
```

[50]:
```
cluster_data_annot.isnull().sum()
```

[50]:
```
GEBIET_NAME                            0
INDIKATOR_JAHR                         0
unemployment_capita                  224
nrw_svp                             3428
taxable_income_natural                 0
students_per_1000pop                 599
exp_environmental_protection_capita    0
garbage_capita                         0
dtype: int64
```

[51]:
```
cluster_data_annot = cluster_data_annot.dropna(subset=['unemployment_capita'])
cluster_data_annot = cluster_data_annot.dropna(subset=['students_per_1000pop'])
cluster_data_annot = cluster_data_annot.reset_index(drop=True)
cluster_data_annot.shape
```

[51]:
```
(3702, 8)
```

[52]:
```
cluster_data_annot.isnull().sum()
```

[52]:
```
GEBIET_NAME                            0
INDIKATOR_JAHR                         0
unemployment_capita                    0
nrw_svp                             2730
taxable_income_natural                 0
students_per_1000pop                   0
exp_environmental_protection_capita    0
garbage_capita                         0
dtype: int64
```

[53]:
```
def fill_nans_with_next_value(df):
    last_values = {}  # Dictionary to store the last value used for each␣
 ↪"GEBIET_NAME"
```

```
    for i in range(len(df)):
        gebiet_name = df.loc[i, 'GEBIET_NAME']

        if pd.isna(df.loc[i, 'nrw_svp']):
            # If the last value for this "GEBIET_NAME" is already known, use it
            if gebiet_name in last_values:
                df.loc[i, 'nrw_svp'] = last_values[gebiet_name]
                continue

            # Find all non-NaN values for this "GEBIET_NAME"
            available_values = df[df['GEBIET_NAME'] == gebiet_name]['nrw_svp'].
↪dropna()

            # If there's only one value available, use it for all NaNs for this␣
↪"GEBIET_NAME"
            if len(available_values) == 1:
                value = available_values.iloc[0]
                df.loc[df['GEBIET_NAME'] == gebiet_name, 'nrw_svp'] = value
                last_values[gebiet_name] = value
            # If there's more than one value, find the last one and use it if␣
↪available; else find the next one
            else:
                last_index = available_values[available_values.index < i].
↪last_valid_index()
                if last_index is not None:
                    last_value = df.loc[last_index, 'nrw_svp']
                    df.loc[i, 'nrw_svp'] = last_value
                    last_values[gebiet_name] = last_value
                else:
                    next_index = available_values[available_values.index > i].
↪first_valid_index()
                    if next_index is not None:
                        next_value = df.loc[next_index, 'nrw_svp']
                        df.loc[i, 'nrw_svp'] = next_value
                        last_values[gebiet_name] = next_value
        else:
            # If the value is not NaN, update the last known value for this␣
↪"GEBIET_NAME"
            last_values[gebiet_name] = df.loc[i, 'nrw_svp']

    return df
```

[54]:
```
fill_nans_with_next_value(cluster_data_annot)
```

[54]:
```
              GEBIET_NAME  INDIKATOR_JAHR  unemployment_capita  nrw_svp  \
0     Adlikon (bis 2022)            1999             0.002218     69.2
1     Adlikon (bis 2022)            2001             0.002542     69.2
2     Adlikon (bis 2022)            2002             0.006010     69.2
3     Adlikon (bis 2022)            2003             0.008878     60.8
4     Adlikon (bis 2022)            2004             0.017271     60.8
...                  ...             ...                  ...      ...
3697              Zürich            2017             0.022282     18.1
3698              Zürich            2018             0.016261     18.1
3699              Zürich            2019             0.013241     13.7
3700              Zürich            2020             0.019032     13.7
3701              Zürich            2021             0.017320     13.7

      taxable_income_natural  students_per_1000pop  \
0                    46364.0                  87.0
1                    49026.0                  83.2
2                    48809.0                  76.3
3                    46763.0                  91.8
4                    47315.0                  92.1
...                      ...                   ...
3697                 62355.0                  26.3
3698                 63315.0                  26.3
3699                 63622.0                  26.7
3700                 64779.0                  26.9
3701                 64871.0                  27.6

      exp_environmental_protection_capita  garbage_capita
0                                    76.0        0.170648
1                                    73.0        0.193220
2                                    74.0        0.213689
3                                    63.0        0.189280
4                                    82.0        0.196891
...                                   ...             ...
3697                                 66.0        0.238119
3698                                 62.0        0.234074
3699                                113.0        0.234330
3700                                117.0        0.231343
3701                                121.0        0.225057

[3702 rows x 8 columns]
```

[55]:
```
cluster_data_annot.isnull().sum()
```

[55]:
```
GEBIET_NAME                          0
INDIKATOR_JAHR                       0
unemployment_capita                  0
nrw_svp                              0
```

```
taxable_income_natural                0
students_per_1000pop                  0
exp_environmental_protection_capita   0
garbage_capita                        0
dtype: int64
```

[56]:
```
# SAVING A BACKUP CSV FILE WITH TRANSFORMED AND CLEANED PIVOT DATA FOR CLUSTERING

## Uncomment to run:
## cluster_data_annot.to_csv(CSV_PATH + "cluster_data_annot.csv", index=False)
```

[57]:
```
####################
```

[58]:
```
# CLUSTERING DATA

## Creating different versions of the data frame without annotations - i.e.␣
↪without year and region

cluster_data_kmeans_2D = cluster_data_kmeans_3D = cluster_data_hierarchical =␣
↪cluster_data_pca_test = cluster_data_annot[['unemployment_capita',
      'nrw_svp', 'taxable_income_natural',␣
↪'exp_environmental_protection_capita',
      'garbage_capita', 'students_per_1000pop']]
```

[59]:
```
## Packages for Clustering
from sklearn.pipeline import make_pipeline
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import numpy as np
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import AgglomerativeClustering
```

[60]:
```
## PCA Test
scaler = StandardScaler()
pca = PCA()
pipeline = make_pipeline(scaler, pca)

# Fit the pipeline to 'samples'
pipeline.fit(cluster_data_pca_test)
features = range(pca.n_components_)
plt.bar(features, pca.explained_variance_ratio_)
plt.xlabel('PCA feature')
plt.ylabel('explained variance ratio')
plt.title('Abbildung C: Hauptkomponentenanalyse')
```

```
plt.xticks(features)
plt.show()
print(pca.explained_variance_ratio_)
```

```
[0.27091614 0.20891997 0.18880437 0.12745415 0.11392125 0.08998412]
```

[61]:
```
## K-Means Clustering / PCA(n_components=3)

# Create scaler: scaler
scaler = StandardScaler()

# Create a PCA instance: pca
pca = PCA(n_components=3)

# Create pipeline: pipeline
pipeline = make_pipeline(scaler, pca)

# Fit the pipeline to 'samples'
pipeline.fit(cluster_data_kmeans_3D)

# Transform the data
pca_features = pipeline.transform(cluster_data_kmeans_3D)

# Range of clusters
range_n_clusters = range(2, 11)
silhouette_scores = []
best_score_kmeans_3d = -1
best_n_clusters_kmeans_3d = 0

# Define the number of runs for each number of clusters
n_runs = 10

# Create an empty list to store the best models for each number of clusters
best_models = []

# Loop through different number of clusters
for n_clusters in range_n_clusters:
    best_sse = np.inf
    best_model = None

    # Perform K-means multiple times and store the results
    for _ in range(n_runs):
        kmeans = KMeans(n_clusters=n_clusters, init='random', n_init=1)
```

```
        kmeans.fit(pca_features)
        sse = kmeans.inertia_
        if sse < best_sse:
            best_sse = sse
            best_model = kmeans

    # Predict labels and compute silhouette score for the best model
    labels = best_model.predict(pca_features)
    score = silhouette_score(pca_features, labels)
    silhouette_scores.append(score)

    if score > best_score_kmeans_3d:
        best_score_kmeans_3d = score
        best_n_clusters_kmeans_3d = n_clusters

    best_models.append(best_model)
    print(f"For n_clusters = {n_clusters}, best SSE is {best_sse}, silhouette␣
 ↪score is {score}")

# Plot silhouette scores
fig_sil_kmeans_3d, ax1 = plt.subplots()
ax1.plot(range_n_clusters, silhouette_scores, marker='o')
ax1.set_title('Abbildung 5: Silhouette Scores für verschiedene Anzahl von␣
 ↪Clustern')
ax1.set_xlabel('Number of Clusters')
ax1.set_ylabel('Silhouette Score')

# Use the best model for the optimal number of clusters for visualization
best_model = best_models[np.argmax(silhouette_scores)]
labels = best_model.predict(pca_features)

fig_kmeans_3d = plt.figure()
ax2 = fig_kmeans_3d.add_subplot(111, projection='3d')
ax2.scatter(pca_features[:, 0], pca_features[:, 1], pca_features[:, 2],␣
 ↪c=labels, cmap='viridis', alpha=0.5)
ax2.set_title('Abbildung 6: Visualisierung Clustering im 3D-Raum')

plt.show()
```

```
For n_clusters = 2, best SSE is 11187.49471044329, silhouette score is
0.25002170682597896
For n_clusters = 3, best SSE is 8839.79407456276, silhouette score is
0.2660151611678685
For n_clusters = 4, best SSE is 7047.436455927353, silhouette score is
0.27593163856502023
For n_clusters = 5, best SSE is 6268.443941702041, silhouette score is
0.2514541814822931
```

```
For n_clusters = 6, best SSE is 5679.632962986525, silhouette score is
0.257278047027093
For n_clusters = 7, best SSE is 5138.682867333043, silhouette score is
0.2553519064270284
For n_clusters = 8, best SSE is 4777.84398242756, silhouette score is
0.24037081015902906
For n_clusters = 9, best SSE is 4468.656045639569, silhouette score is
0.23840295788442475
For n_clusters = 10, best SSE is 4187.68391986268, silhouette score is
0.2405927809105781
```

[62]:
```
## Adding labels back to their DataFrame

cluster_data_kmeans_3D['cluster_kmeans_3D'] = labels
cluster_data_annot['cluster_kmeans_3D'] = labels
```

```
/var/folders/7q/41d5qrlx2nb8wlrcz2s09qwh0000gn/T/ipykernel_5630/2520949439.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  cluster_data_kmeans_3D['cluster_kmeans_3D'] = labels
```

[63]:
```
## K-Means Clustering / PCA(n_components=2)


scaler = StandardScaler()

# Create a PCA instance: pca
pca = PCA(n_components=2)

# Create pipeline: pipeline
pipeline = make_pipeline(scaler, pca)

# Fit the pipeline to 'samples'
pipeline.fit(cluster_data_kmeans_2D)

# Transform the data
```

```
pca_features = pipeline.transform(cluster_data_kmeans_2D)

# Range of clusters
range_n_clusters = range(2, 11)
silhouette_scores = []
best_score_kmeans_2d = -1
best_n_clusters_kmeans_2d = 0

# Define the number of runs for each number of clusters
n_runs = 10

# Create an empty list to store the best models for each number of clusters
best_models = []

# Loop through different number of clusters
for n_clusters in range_n_clusters:
    best_sse = np.inf
    best_model = None

    # Perform K-means multiple times and store the results
    for _ in range(n_runs):
        kmeans = KMeans(n_clusters=n_clusters, init='random', n_init=1)
        kmeans.fit(pca_features)
        sse = kmeans.inertia_
        if sse < best_sse:
            best_sse = sse
            best_model = kmeans

    # Predict labels and compute silhouette score for the best model
    labels = best_model.predict(pca_features)
    score = silhouette_score(pca_features, labels)
    silhouette_scores.append(score)

    if score > best_score_kmeans_2d:
        best_score_kmeans_2d = score
        best_n_clusters_kmeans_2d = n_clusters

    best_models.append(best_model)
    print(f"For n_clusters = {n_clusters}, best SSE is {best_sse}, silhouette␣
 ↪score is {score}")

# Plot silhouette scores
fig_sil_kmeans_2d, ax1 = plt.subplots()
ax1.plot(range_n_clusters, silhouette_scores, marker='o')
ax1.set_title('Abbildung 7: Silhouette Scores für verschiedene Anzahl von␣
 ↪Clustern')
ax1.set_xlabel('Number of Clusters')
```

```
ax1.set_ylabel('Silhouette Score')

# Use the best model for the optimal number of clusters for visualization
best_model = best_models[np.argmax(silhouette_scores)]
labels = best_model.predict(pca_features)

fig_kmeans_2d, ax2 = plt.subplots()
scatter = ax2.scatter(pca_features[:, 0], pca_features[:, 1], c=labels,␣
↪cmap='viridis', alpha=0.5)
ax2.set_title('Abbildung 8: Visualisierung Clustering im 2D-Raum')
plt.show()
```

```
For n_clusters = 2, best SSE is 7783.2794823166605, silhouette score is
0.4482015182801492
For n_clusters = 3, best SSE is 4344.7602998554075, silhouette score is
0.5193124222236909
For n_clusters = 4, best SSE is 3317.4382281085, silhouette score is
0.4121353380199387
For n_clusters = 5, best SSE is 2861.0094664788835, silhouette score is
0.3845365481785845
For n_clusters = 6, best SSE is 2533.039289436201, silhouette score is
0.3754972975568304
For n_clusters = 7, best SSE is 2247.899187125553, silhouette score is
0.3360285710686103
For n_clusters = 8, best SSE is 1971.8398945293584, silhouette score is
0.35985174293384464
For n_clusters = 9, best SSE is 1804.5090267826363, silhouette score is
0.36356879673851333
For n_clusters = 10, best SSE is 1657.9162573693707, silhouette score is
0.35566836069815
```

[64]:
```
## Adding labels back to their DataFrame

cluster_data_kmeans_2D['cluster_kmeans_2D'] = labels
cluster_data_annot['cluster_kmeans_2D'] = labels
```

```
/var/folders/7q/41d5qrlx2nb8wlrcz2s09qwh0000gn/T/ipykernel_5630/4125727771.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  cluster_data_kmeans_2D['cluster_kmeans_2D'] = labels
```

[65]:
```
## Hierarchical Clustering / PCA(n_components=3)

# Create scaler: scaler
scaler = StandardScaler()

# Create a PCA instance: pca
pca = PCA(n_components=3)

# Create pipeline: pipeline
pipeline = make_pipeline(scaler, pca)

# Fit the pipeline to 'samples'
pipeline.fit(cluster_data_hierarchical)

# Transform the data
pca_features = pipeline.transform(cluster_data_hierarchical)

range_n_clusters = range(2, 11)
best_score_hierarchical = -1
best_n_clusters_hierarchical = 0
silhouette_scores = []

# Loop through different number of clusters
for n_clusters in range_n_clusters:
    agg_clustering = AgglomerativeClustering(n_clusters=n_clusters)
    labels = agg_clustering.fit_predict(pca_features)
    score = silhouette_score(pca_features, labels)
    silhouette_scores.append(score)

    print(f"For n_clusters = {n_clusters}, silhouette score is {score}")

    if score > best_score_hierarchical:
        best_score_hierarchical = score
        best_n_clusters_hierarchical = n_clusters

# Perform hierarchical clustering with the best number of clusters
agg_clustering = AgglomerativeClustering(n_clusters=best_n_clusters_hierarchical)
labels = agg_clustering.fit_predict(pca_features)

# Plot silhouette scores
fig_sil_hierarchical, ax1 = plt.subplots()
ax1.plot(range_n_clusters, silhouette_scores, marker='o')
```

```
ax1.set_title('Abbildung D: Silhouette Scores für verschiedene Anzahl von␣
 ↪Clustern')
ax1.set_xlabel('Number of Clusters')
ax1.set_ylabel('Silhouette Score')

# Plot the clusters in 3D
fig_hierarchical = plt.figure()
ax = fig_hierarchical.add_subplot(111, projection='3d')
ax.set_title('Abbildung E: Visualisierung Hierarchisches Clustering')
scatter = ax.scatter(pca_features[:, 0], pca_features[:, 1], pca_features[:, 2],␣
 ↪c=labels, cmap='viridis', alpha=0.5)

print(f"Best silhouette score is {best_score_hierarchical} for n_clusters =␣
 ↪{best_n_clusters_hierarchical}")
```

```
For n_clusters = 2, silhouette score is 0.447297846578285
For n_clusters = 3, silhouette score is 0.4942161301730251
For n_clusters = 4, silhouette score is 0.4541955387246892
For n_clusters = 5, silhouette score is 0.4398599955226309
For n_clusters = 6, silhouette score is 0.38877171409202926
For n_clusters = 7, silhouette score is 0.30581847965207687
For n_clusters = 8, silhouette score is 0.2596957623851843
For n_clusters = 9, silhouette score is 0.2267133594684032
For n_clusters = 10, silhouette score is 0.22673255717136762
Best silhouette score is 0.4942161301730251 for n_clusters = 3
```

[66]:
```
## Adding labels back to their DataFrame

cluster_data_hierarchical['cluster_hierarchical'] = labels
cluster_data_annot['cluster_hierarchical'] = labels
```

```
/var/folders/7q/41d5qrlx2nb8wlrcz2s09qwh0000gn/T/ipykernel_5630/69914466.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  cluster_data_hierarchical['cluster_hierarchical'] = labels
```

[67]:
```
## Distribution of clusters across the regions - Creating a function

### For each 'GEBIET_NAME', the function identifies the cluster with the highest␣
 ↪count (i.e., the maximum),
### resulting in a final DataFrame "region_distributed" that contains the␣
 ↪"GEBIET_NAME" and
### the most common cluster label from each of the three clustering methods.

def distribute_clusters(cluster_column):
    cluster_distribution = cluster_data_annot.groupby([cluster_column,␣
 ↪'GEBIET_NAME']).size().reset_index(name='count')
    pivot_distribution = cluster_distribution.pivot(index='GEBIET_NAME',␣
 ↪columns=cluster_column, values='count').fillna(0)
    pivot_distribution['max_cluster'] = pivot_distribution.idxmax(axis=1)
    region_distributed = pivot_distribution['max_cluster'].reset_index()
    region_distributed.columns = ['GEBIET_NAME', cluster_column]
    return region_distributed

# For each cluster type
region_distributed_3D = distribute_clusters('cluster_kmeans_3D')
region_distributed_2D = distribute_clusters('cluster_kmeans_2D')
region_distributed_hierarchical = distribute_clusters('cluster_hierarchical')

# Merging all three dataframes on 'GEBIET_NAME'
region_distributed = region_distributed_3D.merge(region_distributed_2D,␣
 ↪on='GEBIET_NAME').merge(region_distributed_hierarchical, on='GEBIET_NAME')

# Display the first few rows of the DataFrame
print(region_distributed.head())
```

```
          GEBIET_NAME  cluster_kmeans_3D  cluster_kmeans_2D  \
0  Adlikon (bis 2022)                  1                  2
1            Adliswil                  2                  1
2            Aesch ZH                  0                  0
3         Aeugst a.A.                  0                  0
4      Affoltern a.A.                  2                  1

   cluster_hierarchical
0                     2
1                     0
2                     1
3                     0
4                     0
```

[68]:
```
## Geographical visualisation of the clustering

from matplotlib.colors import ListedColormap
```

```
import matplotlib.patches as mpatches
import geopandas as gpd


# Define a function to plot a map for a specific cluster column
def plot_map(data, cluster_column, ax, title):
    # Get the number of unique clusters
    n_clusters = data[cluster_column].nunique()

    # Generate a color palette with as many colors as there are clusters
    colors = sns.color_palette("pastel", max(data[cluster_column].dropna().
 ↪astype(int).unique())+1)

    # Define a colormap with specific colors
    cmap = ListedColormap(colors)

    # Plot the map, coloring by the cluster without the default legend
    data.plot(column=cluster_column, cmap=cmap, legend=False, ax=ax)

    # Create custom legend
    legend_labels = [f"Cluster {i}" for i in range(n_clusters)]
    legend_handles = [mpatches.Patch(color=color) for color in colors[:
 ↪n_clusters]]
    ax.legend(handles=legend_handles, labels=legend_labels)

    # Set title
    ax.set_title(title)

    # Turn off the axis
    ax.set_axis_off()

# Load the map data
map_data = gpd.read_file(CSV_PATH + 'UP_GEMEINDEN_SEEN_F.shp')

# Merge with the clustering results
merged_data = map_data.merge(region_distributed, left_on='GEMEINDENA',␣
 ↪right_on='GEBIET_NAME', how='left')

# Create a figure with 3 subplots
fig, axes = plt.subplots(1, 3, figsize=(20, 10))

# Plot each cluster type in a separate subplot
plot_map(merged_data, 'cluster_kmeans_3D', axes[0], 'K-Means 3D Clusters')
plot_map(merged_data, 'cluster_kmeans_2D', axes[1], 'K-Means 2D Clusters')
plot_map(merged_data, 'cluster_hierarchical', axes[2], 'Hierarchical Clusters')

# Add a centered title above all subplots
```

```
fig.suptitle('Abbildung 9: Geografische Visualisierung der Ergebnisse der␣
 ↪Clusterverfahren')

plt.tight_layout(rect=[0, 0.03, 1, 0.95])  # Adjust the layout to make room for␣
 ↪the title
plt.show()
```

[69]:
```
## Analysis of the clustering

pairplot = sns.pairplot(data=cluster_data_annot, kind="scatter",␣
 ↪vars=['unemployment_capita', 'nrw_svp',
       'students_per_1000pop', 'exp_environmental_protection_capita',
       'garbage_capita'], hue="cluster_kmeans_2D", plot_kws={'alpha': 0.2})


pairplot.fig.suptitle("Abbildung F: Pairplot ausgewählter Features (k-Means,␣
 ↪2D)", y=1.02)

plt.show()
```

[70]:
```
# REGRESSION ANALYSIS
from linearmodels.panel import PanelOLS

## Data preparation
regression_data = cluster_data_annot.set_index(["GEBIET_NAME", "INDIKATOR_JAHR"])
regression_data.head()
regression_data.shape
```

[70]:
```
(3702, 9)
```

[71]:
```
regression_data.dtypes
```

[71]:
```
unemployment_capita                 float64
nrw_svp                             float64
taxable_income_natural              float64
students_per_1000pop                float64
exp_environmental_protection_capita float64
garbage_capita                      float64
cluster_kmeans_3D                     int32
```

```
cluster_kmeans_2D                    int32
cluster_hierarchical                 int64
dtype: object
```

[72]:
```
regression_data.columns
```

[72]:
```
Index(['unemployment_capita', 'nrw_svp', 'taxable_income_natural',
       'students_per_1000pop', 'exp_environmental_protection_capita',
       'garbage_capita', 'cluster_kmeans_3D', 'cluster_kmeans_2D',
       'cluster_hierarchical'],
      dtype='object')
```

[73]:
```
## Transforming the Cluster Labels Columns into Dummy Variables
regression_data = pd.get_dummies(regression_data, columns=['cluster_kmeans_3D',␣
↪'cluster_kmeans_2D', 'cluster_hierarchical'], drop_first=True)
regression_data.columns
```

[73]:
```
Index(['unemployment_capita', 'nrw_svp', 'taxable_income_natural',
       'students_per_1000pop', 'exp_environmental_protection_capita',
       'garbage_capita', 'cluster_kmeans_3D_1', 'cluster_kmeans_3D_2',
       'cluster_kmeans_3D_3', 'cluster_kmeans_2D_1', 'cluster_kmeans_2D_2',
       'cluster_hierarchical_1', 'cluster_hierarchical_2'],
      dtype='object')
```

[74]:
```
## 2FE Regression

mod = PanelOLS.from_formula(
    "garbage_capita ~ (unemployment_capita + nrw_svp + np.
↪log(exp_environmental_protection_capita) + np.log(taxable_income_natural) +␣
↪students_per_1000pop + (unemployment_capita * np.
↪log(exp_environmental_protection_capita)) + cluster_kmeans_2D_1 +␣
↪cluster_kmeans_2D_2 + cluster_kmeans_3D_1 + cluster_kmeans_3D_2 +␣
↪cluster_hierarchical_1 + cluster_hierarchical_2) + EntityEffects +␣
↪TimeEffects",
    data=regression_data
)


result = mod.fit(cov_type='clustered', cluster_entity=True, cluster_time=True)
result.summary.tables[1]
```

```
/Users/simonsahli-studio/opt/anaconda3/lib/python3.9/site-
packages/pandas/core/arraylike.py:402: RuntimeWarning: invalid value encountered
in log
  result = getattr(ufunc, method)(*inputs, **kwargs)
/Users/simonsahli-studio/opt/anaconda3/lib/python3.9/site-
packages/linearmodels/panel/model.py:1214: MissingValueWarning:
```

```
Inputs contain missing values. Dropping rows with missing observations.
  super().__init__(dependent, exog, weights=weights, check_rank=check_rank)
```

[74]:
```
<class 'statsmodels.iolib.table.SimpleTable'>
```

[75]:
```
## Homoscedasticity?

### Extract residuals and fitted values from the model
residuals = result.resids
fitted = result.fitted_values

### Plotting
plt.figure(figsize=(10,6))
plt.scatter(fitted, residuals, alpha=0.5)
plt.axhline(0, color='red', linestyle='--')
plt.xlabel('Fitted values')
plt.ylabel('Residuals')
plt.title('Abbildung G: Residuals vs. Fitted Values')
plt.show()
```

[76]:
```
## Check whether the error terms follow a normal distribution
import statsmodels.api as sm

residuals = result.resids
sm.qqplot(residuals, line='45', fit=True)
plt.title('Abbildung H: QQ Plot of Residuals')
plt.show()
```

[77]:
```
# R-squared
r_squared = result.rsquared

# RMSE
residuals = result.resids
n = len(residuals)
rmse = (sum(residuals**2) / n)**0.5

print("R-squared:", r_squared)
print("RMSE:", rmse)
```

```
R-squared: 0.2192018940486059
RMSE: 0.01944009542508675
```

[ ]: