

Report

Gruppe D: Nico Gerspach & Jonas Lüttmann

1 Introduction and data

Nachdem wir uns einen Überblick über die möglichen Datenquellen verschafft haben, sind wir zum Entschluss gekommen, uns die Zensus Daten von 2011 genauer anzuschauen. Die Zensus Daten 2011 enthalten unter anderem Angaben über die Erwerbstätigkeit in Deutschland sowie unterschiedlichste soziodemografische Informationen (vgl. [Zensus 2011](#)). Hinsichtlich der Arbeitslosigkeit gibt es diverse Vorurteile und Vermutungen. So wird häufig behauptet, dass der Großteil der Arbeitslosen Personen mit Migrationshintergrund sind oder ein schlechtes Bildungsniveau vorweisen. Die Bundeszentrale für politische Bildung veröffentlichte hierzu einen Bericht, welcher diese zwei Vermutungen sogar bestätigt (vgl. [Arbeitslosenquoten nach Geschlecht und Staatsangehörigkeit](#), 2021), (vgl. [Arbeitslosenquoten nach Bildung und Alter](#), 2021). Diese Informationen waren für uns Anreiz genug, um diese Zusammenhänge zu untersuchen.

Die Fragestellung, welche wir innerhalb dieses Projekts untersuchen, lautet wie folgt:

i Fragestellung

Gibt es einen Zusammenhang zwischen den soziodemografischen Merkmalen und der Arbeitslosenrate?

Die Arbeitslosigkeit im Allgemeinen ist ein Indikator für die Situation auf dem Arbeitsmarkt. Die Arbeitslosenquote kann unter Angabe der “Anzahl Erwerbslosen” sowie der “Anzahl Erwerbstätigen” berechnet werden. Als soziodemografische Merkmale haben wir uns für folgende sechs entschieden und wie folgt definiert (siehe Kapitel 5.1). Der Grund wieso wir uns für die 6 Merkmale entschieden haben war, da aufgrund der Zensus Umfrage die soziodemografischen Merkmale in verschiedene Kategorien aufgeteilt waren. Dies sind Informationen auf Gemeindeebene, wie z.B. Angaben zum Familienstand (Anzahl Personen die ledig sind, verheiratet, usw.), Angaben nach Religionszugehörigkeit (Anzahl Personen die römisch-katholisch, evangelisch oder sonstiges) oder Angaben zum Bildungsniveau (Anzahl Personen ohne beruflichen Abschluss, mindestens eine Lehre, mindestens Hochschulabschluss, usw.). Aufgrund dieser Kategorien entschieden wir uns Quoten zu bilden, die unserer Meinung nach in einer Beziehung zur Arbeitslosenquote stehen können.

1.1 Import relevanter Module

1.2 Import Datensatz

1.3 Data Structure

Um einen ersten groben Überblick zu bekommen, geben wir uns eine Info über unseren Datensatz mit der Pandas-Funktion `pd.info()` aus. Hierbei ist zu sehen, dass 166 Spalten als Datentyp `object` haben und somit auf gemischte Datentypen hindeutet (siehe Kapitel 5.2). In diesem ist ersichtlich, dass manche Spalten einige leere Zellen enthalten. So gibt es einige Spalten mit nur 2187 non-null Werten. Die ersten 10 Zeilen sowie die letzten 10 Zeilen des Datensatz verdeutlichen ebenso, dass nicht alle Werte nutzbar sind (siehe Tabelle 18 & Tabelle 19). Daher führen wir im nächsten Schritt Daten Korrekturen durch, um saubere Datentypen zu haben.

1.4 Data Correction

i Korrekturen

- integer in float umwandeln
- / und - in 0-Werte verwandeln, da diese im engeren Sinne als 0 zählen
- Zahlen in Klammern als normale Zahlen umwandeln

Anschließend erfolgt der Check, ob die Anpassung der Zahlen, auf Basis zweier bekannter Gemeinden mit ursprünglich nicht korrekt formatierten Werten, nun korrekt ist:

	DEM_2.7	DEM_2.10
43	71.0	9.0
44	19.0	0.0

Die definierten predictor variables (siehe Kapitel 5.1) müssen im Folgenden noch berechnet werden:

1.4.1 Data splitting 1

Unter Angabe des Spaltenindex filtern wir den Dataframe, sodass wir nur noch die relevanten Spalten erhalten und kopieren diese Werte in ein neues Dataframe `df_analyse`:

Nicht jede Zelle in unserem Dataframe `df_analyse` enthält numerische Werte, da öfters die Info NaN angezeigt wird.

i Gründe, weshalb NaN vorkommt können folgende sein:

- Daten sind nicht erhoben worden
- Zahlenwert der Erfassung nicht sicher genug
- Aufgrund von Geheimhaltungsverfahren

Die erneute Ausführung des Codes `pd.info()` zeigt, dass nur noch die relevanten Quoten enthalten sind und informative Spalten. Dabei ist ersichtlich, dass manche Spalten deutlich weniger NaN Werte enthalten (siehe Kapitel 5.3)

Da mit NaN Werten nicht gerechnet werden kann müssen diese Zeilen entfernt werden. Mithilfe der `.dropna` Funktion entfernen wir diese Zeilen deren relevanten Spalten NaN Werte enthalten. Im gleichen Schritt wird der Dataframe auf Hierarchie-Ebene **Gemeinde** gefiltert. Das neue Dataframe heißt nun `df_analyse_gemeinde` (Siehe Tabelle 20).

Die Ausführung der Funktion `pd.info()` ergibt folgende Übersicht. Unser nun bereinigter Dataframe enthält 1574 Zeilen, welcher die Basis für die Anwendung der Modelle bildet.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1574 entries, 0 to 11303
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Name                   1574 non-null   category
1   Reg_Hier               1574 non-null   category
2   Arbeitslosenquote     1574 non-null   float64
3   Migrationsquote       1574 non-null   float64
4   Christenquote         1574 non-null   float64
5   Männerquote           1574 non-null   float64
6   Akademikerquote       1574 non-null   float64
```

```

7   Beamtenquote      1574 non-null   float64
8   Singlequote       1574 non-null   float64
dtypes: category(2), float64(7)
memory usage: 452.9 KB

```

1.4.2 Variable List

Im Folgenden werden die Prädikatoren sowie die Outcome-Variable definiert. Zudem werden die Daten für die Prädikatoren sowie die Outcome-Variable definiert.

1.4.3 Data Splitting 2

Für das spätere Modell möchten wir Trainings- und Testdaten. Um die deskriptive und explorative Datenanalyse bereits auf den Trainingsdaten durchzuführen, splitten wir im nächsten Schritt die Daten. Dies machen wir mit der `train_test_split`-Funktion von `scikit-learn`.

1.5 Descriptive Analytics

Die statistischen Werte, die mithilfe der Funktion `.describe` ausgegeben werden, geben ein erstes Gefühl für die bereinigten Daten, welche für die Erstellung der Modelle verwendet werden.

i Die Funktion `.describe` enthält:

- Lagemaße (Mittelwert, Median)
- Streuungsmaße (Standardabweichung, Quartile)

	Arbeitslosenquote	Migrationsquote	Christenquote	Männerquote	Akademikerquote	Beamtenquote	Singlequote
count	1,574.00	1,574.00	1,574.00	1,574.00	1,574.00	1,574.00	1,574.00
mean	4.22	17.76	62.14	48.70	13.77	5.07	52.21
std	2.10	9.63	20.94	0.84	5.85	1.74	3.01
min	0.78	0.85	5.93	45.10	2.09	1.24	44.40
25%	2.80	10.66	58.62	48.19	9.78	3.89	50.25
50%	3.62	17.84	68.01	48.70	12.34	4.89	51.69
75%	5.02	24.11	75.94	49.18	16.24	6.02	53.66
max	16.87	53.98	93.91	54.99	48.00	18.87	66.46

1.6 Explorative Analytics

Um die Verteilung der zugrundeliegenden Daten grafisch darstellen zu können, erstellen wir für jede Variable ein Histogramm. Das Histogramm für die "Christenquote" weist eine linksschiefe, multimodale Verteilung auf. Die "Männerquote" weist eine annähernd symmetrische, unimodale Verteilung auf. Alle weiteren Variablen sind rechtsschief, unimodal verteilt.

```
alt.RepeatChart(...)
```

Zur Visualisierung der Beziehungen zwischen Response- und Predictor Variables haben wir uns für die Anwendung von Streudiagrammen bzw. Scatter Plots entschieden. Jeden Prädiktor haben wir mit der Arbeitslosenquote gegenübergestellt, um erste Erkenntnisse gewinnen zu können. Jedoch ist es anhand der Scatter Plots zunächst schwierig zu erkennen, welcher Prädiktor die höchste Korrelation mit der Arbeitslosigkeit aufweist, da die berechneten Variablen unterschiedliche Werte bzw. Verteilungen auf der X-Achse annehmen und kein direkter Vergleich stattfinden kann.

```
alt.RepeatChart(...)
```

Zu erkennen ist, dass lediglich die Variable “Christenquote” eine moderate Korrelation mit der Arbeitslosenquote aufweist. Zwei weitere Quoten, die noch eine leichte bis moderarte Korrelation haben sind die “Singlequote” sowie die “Migrationsquote”.

Pearson Korrelations-Koeffizienten

Unter Anwendung der Funktion `.corr` werden die Korrelationen berechnet und in Tabellenform ausgegeben.

	Arbeitslosenquote	Migrationsquote	Christenquote	Männerquote	Akademikerquote	Beamtenquote	Singlequote
Arbeitslosenquote	1.000000	-0.273260	-0.653340	-0.231350	-0.121500	-0.257300	0.426140
Migrationsquote	-0.273260	1.000000	0.417650	-0.050460	0.093190	-0.006110	0.093330
Christenquote	-0.653340	0.417650	1.000000	0.142780	-0.248540	0.275360	-0.285430
Männerquote	-0.231350	-0.050460	0.142780	1.000000	-0.284900	-0.065030	-0.287660
Akademikerquote	-0.121500	0.093190	-0.248540	-0.284900	1.000000	0.301710	0.246270
Beamtenquote	-0.257300	-0.006110	0.275360	-0.065030	0.301710	1.000000	-0.014930
Singlequote	0.426140	0.093330	-0.285430	-0.287660	0.246270	-0.014930	1.000000

Die Pearson Korrelations-Koeffizienten bestätigen die oben genannten Vermutungen.

i Korrelationen

- Moderate Korrelation mit der Arbeitslosenquote: “Christenquote”
- Leichte bis moderate Korrelation: “Singlequote”
- Äußerst geringe Korrelation: “Akademikerquote”

2 Methodology

Wie in der Introduction beschrieben haben wir uns aufgrund der Literatur-Recherche wie auch nach Betrachtung des Data Sets für 6 Quoten entschieden, welche die Arbeitslosenquote beeinflussen könnten.

Auf Basis der explorativen Analyse wurde bereits an dieser Stelle beschlossen, nur die “Christenquote”, “Singlequote” und die “Migrationsquote” in die Auswahl für die Modelle aufzunehmen. Da in den Scatterplots linere Zusammenhänge erkennbar sind, wollen wir versuchen, diese mit linearen Modellen zu erklären.

Zu Beginn der jeweiligen Modellierungsprozesse haben wir jeweils das Regressionsmodell ausgewählt. Der nächste Schritt war es die Modelle zu validieren und an die Daten anzupassen. Die Validierung haben wir mit der Cross-Validation umgesetzt und dabei den Mean Squared error pro Fold angeschaut.

Daraufhin haben wir mit der Funktion `reg.fit` die Modelle an die Trainingsdaten angepasst und somit trainiert und den Intercept mit der y-Achse sowie den/die Koeffizienten berechnet.

Der abschließende Schritt war die Evaluierung mit den Test-Daten. Schlussendlich haben wir die Modelle anhand verschiedener Gütemaße (R^2 , MSE, RMSE, MAE) bewertet.

Durch die Peer-Review haben wir das Feedback bekommen, dass wir für die Multiple Regression die Stepwise Selection nutzen könnten, was wir dann im Folgenden auch umgesetzt haben.

Diese grob beschriebenen Schritte wollen wir im Folgenden detaillierter erklären.

2.1 Lineare Regression

2.1.1 Modell-Auswahl

Zunächst werden für die einzelnen Quoten drei Modelle mit jeweils der Linearen Regression definiert. Hier greifen wir auch auf scikit-learn zurück und nehmen `LinearRegression` als Modell.

2.1.2 Training & Validation

Im nächsten Schritt werden die Modelle trainiert und validiert. Dies wird mit der Cross-Validation durchgeführt. Hierbei berechnen wir für jedes Modell den Mean-Squared-Error für je fünf Folds. Hierfür nehmen wir die Funktion `cross_val_score` und visualisieren das zum einen in einer Tabelle und zum anderen als Liniendiagramm. Dabei sehen wir, dass die Mean-Squared-Errors je Fold voneinander abweichen. Grundsätzlich ist der MSE für die "Christenquote" aber immer am geringsten, weshalb wir uns im Rahmen der Linearen Regression auf die "Christenquote" konzentrieren.

```
alt.RepeatChart(...)
```

2.1.3 Fit Model

Im nächsten Schritt möchten wir das Modell an die Trainingsdaten anpassen und den y-Achsenabschnitt sowie die Steigung berechnen. Hierfür nutzen wir die `.fit` Funktion.

Dabei ergeben sich für das lineare Modell mit der "Christenquote" als predictor folgende Werte:

	Name	Coefficient
0	Intercept	8.194
1	slope	-0.065

2.1.4 Evaluation on test set

Im nächsten Schritt evaluieren wir unser Modell mit den Testdaten. Dazu prognostizieren wir y-Werte. Das bedeutet "Arbeitslosenquoten" auf Basis der Testdaten, welche diverse "Christenquoten" darstellen. Dies setzen wir mit der Funktion `.predict` um.

Für die prognostizierten Werte berechnen wir Gütemaße, wie den R-squared (R^2), den Mean-Squared-Error (MSE), den Rooted-Mean-Squared-Error (RMSE) sowie den Mean-Absolute-Error (MAE).

	Christenquote
R^2	0.418
MSE	2.613
RMSE	1.616
MAE	1.194

2.2 Multiple Regression

2.2.1 Modell-Auswahl

Auch für die Multiple Regression nehmen wir von scikit-learn die `LinearRegression` als Modell. Der Unterschied zum vorherigen Abschnitt ist allerdings, dass wir hierbei mehr als eine predictor-Variable berücksichtigen.

2.2.2 Training & Validation

Zuerst nehmen wir alle sechs predictor-Variablen, um das Modell der Multiplen Regression zu trainieren und validieren. Hierfür nutzen wir den gleichen Cross-Validation Ansatz wie bei der Linearen Regression.

	lr_mlg_chr_sin
1	4.19582720662
2	4.1908442708
3	3.45180893842
4	4.41455390783
5	3.95138537967

	lr_mhlgchr_slr_m
1	4.19582720662
2	4.19084427081
3	3.45180893842
4	4.41455390784
5	3.95138537968

Im nächsten Schritt möchten wir die Wrapper-Methoden nutzen, um zu prüfen, ob die Multiple Regression mit weniger Features besser performt und somit leichter verständlich wird. Hierfür nutzen wir von scikit-learn den `SequentialFeatureSelector`. Nachdem wir die Anzahl der ausgewählten Features selektiert haben und den MSE für je fünf Folds angeschaut haben, kamen wir zu der Erkenntnis, dass die Multiple Regression mit der Anzahl von fünf Features am besten performt. Diese Variationen haben wir mit der Forward- und Backward-Selection durchgeführt.

Das Ergebnis der Backward- und Forward-Selection sind jeweils die gleichen fünf Features. Das ist eher untypisch. Obwohl der `SequentialFeatureSelector` bereits eine Cross-Validation mit fünf Folds durchführt, machen wir das für auch für die fünf selektierten Variablen, um die beiden Multiplen Regressionen zu vergleichen.

i Erkenntnis

Hierbei sehen wir, dass die Mean-Squared-Errors für die Multiple Regression mit fünf Features niedriger sind, weshalb wir im weiteren Verlauf diese fünf Features für die Multiple Regression berücksichtigen werden.

	lr_christenquote	lr_mannerquote
1	2.222066544073	2.222066544073
2	2.844484011325	2.844484011325
3	2.189314853467	2.189314853467
4	2.539071708859	2.539071708859
5	2.765169458849	2.765169458849

	Features
1	Christenquote
2	Männerquote
3	Akademikerquote
4	Beamtenquote
5	Singlequote

2.2.3 Fit Model

In diesem Schritt passen wir wieder das Modell an die Trainingsdaten an und berechnen den y-Achsenabschnitt sowie die Steigung.

	Name	Coefficient
0	Intercept	20.179
1	Christenquote	-0.065
2	Männerquote	-0.427
3	Akademikerquote	-0.149
4	Beamtenquote	0.044
5	Singlequote	0.204

2.2.4 Evaluation on test set

Im nächsten Schritt evaluieren wir unser Modell mit den Testdaten. Dazu prognostizieren wir y-Werte. Das bedeutet "Arbeitslosenquoten" auf Basis der Testdaten, welche diverse "Christenquoten" darstellen. Dies setzen wir erneut mit der Funktion `.predict` um.

Für die prognostizierten Werte berechnen wir Gütemaße, wie den R-squared (R2), den Mean-Squared-Error (MSE), den Rooted-Mean-Squared-Error (RMSE) sowie den Mean-Absolute-Error (MAE).

	Christenquote	MultipleRegression
R2	0.418	0.625
MSE	2.613	1.683
RMSE	1.616	1.297
MAE	1.194	0.974

2.3 Lasso Regression

2.3.1 Modell-Auswahl

Bei der Lasso Regression ist es zuerst notwendig, die Variablen zu standardisieren, da Lasso am besten performt, wenn die Features um den Wert 0 zentriert sind. Für diese Standardisierung definieren wir zuerst mit der `StandardScaler().fit`-Funktion unseren Skalierer. Diese scikit-learn Funktion entfernt den Mittelwert und skaliert jedes Feature auf eine Einheitsvarianz. Das wird für jedes Feature getrennt durchgeführt. Im nächsten Schritt transformieren wir unsere Trainings- und Testdaten mit diesem Skalierer. Für die Berechnung des alpha-Wertes für Lasso Regression nehmen wir von scikit-learn die `LassoCV`-Funktion, welche bereits eine Cross-Validation integriert hat.

2.3.2 Training & best alpha

Die `LassoCV`-Funktion hat wie bereits erwähnt, die Cross Validation integriert. Somit ermittelt das Modell für uns den bestmöglichen alpha-Wert, um die Regression durchzuführen. Nachdem wir unser Modell trainiert haben, können wir uns mit `.alpha_` den bestmöglichen alpha-Wert ausgeben lassen:

```
0.00730882363957819
```

2.3.3 Fit Model

Anschließend nutzen wir den besten Wert für alpha, um die Lasso Regression damit durchzuführen. Mit dem besten Wert für alpha, werden folgende Koeffizienten berechnet:

	Name	Coefficient
0	Intercept	4.201
1	Migrationsquote	0.000
2	Christenquote	-1.373
3	Männerquote	-0.353
4	Akademikerquote	-0.856
5	Beamtenquote	0.060
6	Singlequote	0.609

2.3.4 Evaluation on test set

Wie in den vorherigen beiden Modellen evaluieren wir unser Modell mit den Test-Daten und prognostizieren Werte.

Für die prognostizierten Werte berechnen wir Gütemaße, wie den R-squared (R^2), den Mean-Squared-Error (MSE), den Rooted-Mean-Squared-Error (RMSE) sowie den Mean-Absolute-Error (MAE).

	Christenquote	MultipleRegression	Lasso
R2	0.418	0.625	0.625
MSE	2.613	1.683	1.686
RMSE	1.616	1.297	1.298
MAE	1.194	0.974	0.973

3 Results

Mithilfe der durchgeführten Analyse konnte festgestellt werden, dass die stärkste positive Korrelation (nach Pearson) zwischen der Arbeitslosenquote einer Gemeinde und dem Prädiktor "Singlequote" besteht ($r = +0.426$). Die stärkste negative Korrelation besteht zwischen der Arbeitslosenquote einer Gemeinde und dem Prädiktor "Christenquote" ($r = -0.615$). Die schwächste Korrelation, auf Basis unserer Daten, weist die "Akademikerquote" auf ($r = -0.121$).

Das Ergebnis der drei durchgeführten Modelle lautet wie folgt:

	Christenquote	MultipleRegression	Lasso
R2	0.418	0.625	0.625
MSE	2.613	1.683	1.686
RMSE	1.616	1.297	1.298
MAE	1.194	0.974	0.973

Betrachtet man die Ergebnisse in der Tabelle, so fällt auf, dass die Multiple Regression bei fast allen Statistiken am besten abschneidet. Sie hat zum einen den größten R2-Wert, den niedrigsten Mean Squared Error, dadurch folglich auch den niedrigsten Rooted Mean Squared Error. Allein der Mean Absolute Error ist minimal höher als bei der Lasso Regression. Zudem fällt auf, dass die Lasso und Multiple Regression in diesen Statistiken eigentlich genau gleich sind. Der Grund dafür ist, dass der gewählte alpha-Wert sehr nahe an null ist und somit so gut wie keinen Einfluss hat. Zudem berücksichtigt zwar die Lasso-Regression alle predictor-Variablen, allerdings beträgt die Steigung der "Migrationsquote" 0, was bedeutet, dass diese keinen Einfluss hat.

Letztendlich haben wir uns für die Multiple Regression als bestes Modell entschieden, um die Abhängigkeit zwischen den soziodemografischen Merkmalen und der Arbeitslosenquote zu beschreiben. Daher speichern wir das Modell und stellen es im Ordner `./models/` bereit.

Unser Modell beinhaltet alle predictor-Variablen, außer der "Migrationsquote". Diese ist im Zuge der schrittweisen Selektion herausgefallen, da mit ihr das Modell schlechter performt hätte. Der r-squared Wert bedeutet, dass unser Modell 62,5 % der Variabilität in der Arbeitslosenquote abdeckt. Der RMSE zeigt uns, dass die prognostizierte Arbeitslosenquote im Mittel um 1.297% um den tatsächlichen Wert streut. Zudem ist im Mittel der absolute Fehler auch bei 0.974%. Stellt man diese Werte ins Verhältnis zu der Spannweite der Arbeitslosenquote, so sind das Werte, die ins Gewicht fallen. So kann eine prognostizierte Arbeitslosenquote in einer Spannweite von beispielsweise 4%-ca.6% liegen.

Bei der Betrachtung der Koeffizienten der predictor-Variablen, fällt auf, dass die "Christenquote" mit nur -0.065 einen sehr niedrigen Wert hat, obwohl sie am Stärksten mit der Arbeitslosenquote korreliert. Die "Männerquote", die nicht so stark mit der Arbeitslosenquote korreliert, hat hingegen mit -0.427 einen deutlichen stärkeren Einfluss auf das Modell. Nimmt man an, dass alle predictor-Variablen konstant sind und nur die "Männerquote" um eine Einheit steigt, so würde die Arbeitslosenquote um 0.427% abnehmen. Diese Erkenntnisse in Verbindung mit dem Wissen, dass die Koeffizienten sich bei der Zunahme der "Migrationsquote" stark verändern, deutet auf eine starke Multikollinearität zwischen den predictor-Variablen hin. Auffällig ist auch, dass die Korrelation zwischen "Christenquote" und "Migrationsquote" bei ca. 0.41 liegt und damit ein weiterer Hinweis auf Multikollinearität ist. Das beeinträchtigt die Qualität unseres Modells. Dies lässt sich auch schon aus der Korrelationsübersicht erkennen, da hier einige predictor-Variablen untereinander stärker korrelieren als mit der Arbeitslosenquote selbst.

	Name	Coefficient
0	Intercept	20.179
1	Christenquote	-0.065
2	Männerquote	-0.427
3	Akademikerquote	-0.149
4	Beamtenquote	0.044
5	Singlequote	0.204

Zusammenfassend lässt sich sagen, dass unser Modell zwar mehr als die Hälfte der Variabilität in der Arbeitslosenquote erklären kann, allerdings trotzdem nicht aussagekräftig genug ist, um die Auswirkungen der soziodemografischen Merkmale auf die Arbeitslosenquote zu erklären. Dies hängt vor allem auch mit der Multikollinearität der Variablen zusammen.

Folgende Herausforderungen gab es bei der Durchführung des Projekts:

i Herausforderungen:

1. Großer Dataframe, obwohl nur wenige Daten tatsächlich relevant waren.
2. Ein Großteil der Zellen relevanter Spalten war leer, da durch die Zensus Umfrage die Werte entweder nicht ermittelt werden konnten oder zu ungenau waren.
3. Geringe Korrelation zwischen Arbeitslosenquote und den Variablen.

4 Discussion + Conclusion

Grundsätzlich macht es Sinn, dass mehrere soziodemografische Merkmale bei der Anwendung der Modelle berücksichtigt werden, da diese Modelle dann besser performen. Dies war bereits aus der Korrelationsübersicht erkennbar, wie auch später bei der Methodology, als verschiedene Modelle untersucht worden sind.

Aufgrund der durchgeführten Data Corrections wurden die tatsächlich genutzten Zeilen bzw. Gemeinden stark reduziert. Von insgesamt 11.339 Gemeinden wurden schlussendlich nur 1.574 Gemeinden berücksichtigt. Daher wäre es spannend zu wissen, ob die berücksichtigten Gemeinden repräsentativ für ganz Deutschland sind. Hierzu vergleichen wir den original Dataframe mit dem bereinigten Dataframe anhand der berücksichtigten Anzahl Einwohner sowie Anzahl Gemeinden.

4.1 Bereinigter DF

```
Anzahl Einwohner      58797595.0
Anzahl Erwerbspersonen 31235700.0
Anzahl Erwerbstätige   29629470.0
dtype: float64
```

4.2 Original DF

```
Anzahl Einwohner      80209997.0
Anzahl Erwerbspersonen 31235700.0
Anzahl Erwerbstätige   29629470.0
dtype: float64
```

Beim Vergleich der Anzahl Einwohner zwischen originalem DF und bereinigtem DF, fällt auf, dass in allen Gemeinden 80.029.997 Einwohner leben. Der bereinigte DF enthält jedoch nur 58.797.595 Einwohner. Die entspricht ca. 73.47 % der gesamten Einwohnerzahl Deutschlands. Die Summe aller Gemeinden beträgt 11.339 Stück. Bereinigt wurden jedoch nur 1.574 Gemeinden berücksichtigt, was einem Anteil von ca. 13.87 % entspricht.

Zusätzlich untersuchten wir auch noch, wie viele Gemeinden je Bundesland in unserem Projekt tatsächlich berücksichtigt worden sind. Auffällig ist, dass es Bundesländer gibt, bei denen lediglich ein Bruchteil der Gemeinden im bereinigten Dataframe enthalten ist. Z.B. Rheinland-Pfalz, Mecklenburg-Vorpommern, Schleswig-Holstein und Thüringen.

	Bundesland Name	Anzahl Gemeinden Original	Anzahl Gemeinden bereinigt	Anteil Gemeinden berücksichtigt in %
14	Baden- Württemberg	1101	246	22.34
15	Bayern	2056	216	10.51
2	Berlin	1	1	100.00
3	Brandenburg	419	72	17.18
10	Bremen	2	2	100.00
8	Hamburg	1	1	100.00
12	Hessen	426	166	38.97

	Bundesland Name	Anzahl Gemeinden Original	Anzahl Gemeinden bereinigt	Anteil Gemeinden berücksichtigt in %
4	Mecklenburg- Vorpommern	808	24	2.97
9	Niedersachsen	1024	205	20.02
11	Nordrhein- Westfalen	396	342	86.36
13	Rheinland-Pfalz	2306	45	1.95
1	Saarland	52	40	76.92
5	Sachsen	470	68	14.47
6	Sachsen-Anhalt	219	59	26.94
0	Schleswig-Holstein	1116	54	4.84
7	Thüringen	942	33	3.50

i Erkenntnis

Dies lässt darauf schliessen, dass in nur 13.87% der Gemeinden Deutschlands, 73.47% der Bevölkerung lebt. Somit wurden für die Untersuchung der Fragestellung überwiegend bevölkerungsreiche Gemeinden berücksichtigt, da kleine Gemeinden tendenziell weniger Angaben machten und somit herausgefiltert wurden. Dadurch, dass auch der Anteil der Gemeinden innerhalb der Bundesländer unterschiedlich hoch ausfällt, ist es fragwürdig, ob die berücksichtigten Daten das gesamte Land gut widerspiegeln.

Es ist in Zukunft hilfreich den Dataframe bereits bei Projektbeginn auf Vollständigkeit der relevanten Variablen zu prüfen. Sollten viele Werte fehlen, wäre es sinnvoll, dass Methoden angewendet werden, die fehlende Werte durch berechnete Werte ersetzen können.

Zudem sollte das Modell auch näher auf die Multikollinearität geprüft werden, da dies die Qualität des Modells stark beeinträchtigt.

5 Appendix

5.1 Appendix Predictor Variables

	Quote	Berechnung
Variable		
Migrationshintergrund	Migrationsquote	(Anzahl Personen mit Migrationshintergrund / Anzahl Personen insgesamt)
Religionszugehörigkeit	Christenquote*	(Römisch-katholische Kirche + Evangelische Kirche) / Bevölkerung nach Religion gesamt
Geschlecht	Männerquote	(Anzahl Männer / Einwohner gesamt)
Bildungsniveau	Akademikerquote**	(Fach- oder Berufsakademie + FH-Abschluss + Hochschulabschluss + Promotion) / höchster beruflicher Abschluss insgesamt
Stellung im Beruf	Beamtenquote	(Anzahl Beamter / Erwerbstätige insgesamt)
Familienstand	Singlequote***	(Anzahl Lediger + Verwitwete + Geschiedene + eingetragene Lebenspartnerschaft aufgehoben + Eingetragener Lebenspartner/- in verstorben + ohne Angaben) / Familienstand gesamt

5.2 Appendix df_bevoelkerung

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12544 entries, 0 to 12543
Columns: 223 entries, AGS_12 to BIL_5.8
dtypes: category(8), float64(41), int64(8), object(166)
memory usage: 21.4+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12544 entries, 0 to 12543
Data columns (total 223 columns):
#   Column          Non-Null Count  Dtype
---  -
0   AGS_12          12544 non-null  category
1   RS_Land         12544 non-null  category
2   RS_RB_NUTS2    12527 non-null  category
3   RS_Kreis       12501 non-null  category
4   RS_VB          12089 non-null  category
5   RS_Gem         11339 non-null  category
6   Name           12544 non-null  category
7   Reg_Hier       12544 non-null  category
8   AEWZ           12544 non-null  int64
9   DEM_1.1        12544 non-null  int64
10  DEM_1.2        12544 non-null  object

```

11	DEM_1.3	12544	non-null	object
12	DEM_2.1	12544	non-null	int64
13	DEM_2.2	12544	non-null	object
14	DEM_2.3	12544	non-null	object
15	DEM_2.4	12544	non-null	object
16	DEM_2.5	12544	non-null	object
17	DEM_2.6	12544	non-null	object
18	DEM_2.7	12544	non-null	object
19	DEM_2.8	12544	non-null	object
20	DEM_2.9	12544	non-null	object
21	DEM_2.10	12544	non-null	object
22	DEM_2.11	12544	non-null	object
23	DEM_2.12	12544	non-null	object
24	DEM_2.13	12544	non-null	object
25	DEM_2.14	12544	non-null	object
26	DEM_2.15	12544	non-null	object
27	DEM_2.16	12544	non-null	object
28	DEM_2.17	12544	non-null	object
29	DEM_2.18	12544	non-null	object
30	DEM_2.19	12544	non-null	object
31	DEM_2.20	12544	non-null	object
32	DEM_2.21	12544	non-null	object
33	DEM_2.22	12544	non-null	object
34	DEM_2.23	12544	non-null	object
35	DEM_2.24	12544	non-null	object
36	DEM_2.25	12544	non-null	object
37	DEM_2.26	12544	non-null	object
38	DEM_2.27	12544	non-null	object
39	DEM_3.1	12544	non-null	int64
40	DEM_3.2	12544	non-null	object
41	DEM_3.3	12544	non-null	object
42	DEM_3.4	12544	non-null	object
43	DEM_3.5	12544	non-null	object
44	DEM_3.6	12544	non-null	object
45	DEM_3.7	12544	non-null	object
46	DEM_3.8	12544	non-null	object
47	DEM_3.9	12544	non-null	object
48	DEM_3.10	12544	non-null	object
49	DEM_3.11	12544	non-null	object
50	DEM_3.12	12544	non-null	object
51	DEM_3.13	12544	non-null	object
52	DEM_3.14	12544	non-null	object
53	DEM_3.15	12544	non-null	object
54	DEM_3.16	12544	non-null	object
55	DEM_3.17	12544	non-null	object
56	DEM_3.18	12544	non-null	object
57	DEM_3.19	12544	non-null	object
58	DEM_3.20	12544	non-null	object
59	DEM_3.21	12544	non-null	object
60	DEM_3.22	12544	non-null	object
61	DEM_3.23	12544	non-null	object
62	DEM_3.24	12544	non-null	object
63	DEM_3.25	12544	non-null	object
64	DEM_3.26	12544	non-null	object

65	DEM_3.27	12544	non-null	object
66	DEM_3.28	12544	non-null	object
67	DEM_3.29	12544	non-null	object
68	DEM_3.30	12544	non-null	object
69	DEM_4.1	12544	non-null	int64
70	DEM_4.2	12544	non-null	object
71	DEM_4.3	12544	non-null	object
72	DEM_4.4	12544	non-null	object
73	DEM_4.5	12544	non-null	object
74	DEM_4.6	12544	non-null	object
75	DEM_4.7	12544	non-null	object
76	DEM_4.8	12544	non-null	object
77	DEM_4.9	12544	non-null	object
78	DEM_4.10	12544	non-null	object
79	DEM_4.11	12544	non-null	object
80	DEM_4.12	12544	non-null	object
81	DEM_4.13	12544	non-null	object
82	DEM_4.14	12544	non-null	object
83	DEM_4.15	12544	non-null	object
84	DEM_4.16	12544	non-null	object
85	DEM_4.17	12544	non-null	object
86	DEM_4.18	12544	non-null	object
87	DEM_4.19	12544	non-null	object
88	DEM_4.20	12544	non-null	object
89	DEM_4.21	12544	non-null	object
90	DEM_4.22	12544	non-null	object
91	DEM_4.23	12544	non-null	object
92	DEM_4.24	12544	non-null	object
93	DEM_4.25	12544	non-null	object
94	DEM_4.26	12544	non-null	object
95	DEM_4.27	12544	non-null	object
96	DEM_4.28	12544	non-null	object
97	DEM_4.29	12544	non-null	object
98	DEM_4.30	12544	non-null	object
99	DEM_4.31	12544	non-null	object
100	DEM_4.32	12544	non-null	object
101	DEM_4.33	12544	non-null	object
102	DEM_4.34	12544	non-null	object
103	DEM_4.35	12544	non-null	object
104	DEM_4.36	12544	non-null	object
105	DEM_5.1	12544	non-null	int64
106	DEM_5.2	12544	non-null	object
107	DEM_5.3	12544	non-null	object
108	DEM_5.4	12544	non-null	object
109	DEM_5.5	12544	non-null	object
110	DEM_5.6	12544	non-null	object
111	DEM_5.7	12544	non-null	object
112	DEM_6.1	12544	non-null	int64
113	DEM_6.2	12544	non-null	object
114	DEM_6.3	12544	non-null	object
115	DEM_6.4	12544	non-null	object
116	DEM_6.5	12544	non-null	object
117	DEM_6.6	12544	non-null	object
118	DEM_6.7	12544	non-null	object

119	REL_1.1	12544 non-null	int64
120	REL_1.2	12544 non-null	object
121	REL_1.3	12544 non-null	object
122	REL_1.4	12544 non-null	object
123	MIG_1.1	2187 non-null	float64
124	MIG_1.2	2187 non-null	float64
125	MIG_1.3	2187 non-null	object
126	MIG_1.4	2187 non-null	object
127	MIG_1.5	2187 non-null	object
128	MIG_1.6	2187 non-null	object
129	MIG_1.7	2187 non-null	object
130	MIG_1.8	2187 non-null	object
131	MIG_1.9	2187 non-null	object
132	MIG_1.10	2187 non-null	object
133	MIG_1.11	2187 non-null	object
134	MIG_2.1	2187 non-null	object
135	MIG_2.2	2187 non-null	object
136	MIG_2.3	2187 non-null	object
137	MIG_2.4	2187 non-null	object
138	MIG_2.5	2187 non-null	object
139	MIG_2.6	2187 non-null	object
140	MIG_2.7	2187 non-null	object
141	MIG_2.8	2187 non-null	object
142	MIG_3.1	2187 non-null	object
143	MIG_3.2	2187 non-null	object
144	MIG_3.3	2187 non-null	object
145	MIG_3.4	2187 non-null	object
146	MIG_3.5	2187 non-null	object
147	ERW_1.1	2187 non-null	float64
148	ERW_1.2	2187 non-null	float64
149	ERW_1.3	2187 non-null	float64
150	ERW_1.4	2187 non-null	float64
151	ERW_1.5	2187 non-null	float64
152	ERW_1.6	2187 non-null	float64
153	ERW_1.7	2187 non-null	float64
154	ERW_1.8	2187 non-null	float64
155	ERW_1.9	2187 non-null	float64
156	ERW_1.10	2186 non-null	object
157	ERW_1.11	2187 non-null	object
158	ERW_1.12	2187 non-null	object
159	ERW_1.13	2187 non-null	float64
160	ERW_1.14	2187 non-null	float64
161	ERW_1.15	2187 non-null	float64
162	ERW_2.1	2187 non-null	float64
163	ERW_2.2	2187 non-null	float64
164	ERW_2.3	2187 non-null	object
165	ERW_2.4	2187 non-null	object
166	ERW_2.5	2187 non-null	object
167	ERW_2.6	2187 non-null	object
168	ERW_3.1	2187 non-null	float64
169	ERW_3.2	2187 non-null	object
170	ERW_3.3	2187 non-null	float64
171	ERW_3.4	2187 non-null	float64
172	ERW_3.5	2187 non-null	float64

173	ERW_3.6	2187	non-null	float64
174	ERW_3.7	2187	non-null	object
175	ERW_3.8	2187	non-null	float64
176	ERW_3.9	2187	non-null	object
177	ERW_3.10	2187	non-null	object
178	ERW_3.11	2187	non-null	object
179	ERW_4.1	2187	non-null	float64
180	ERW_4.2	2187	non-null	object
181	ERW_4.3	2187	non-null	object
182	ERW_4.4	2187	non-null	object
183	ERW_4.5	2146	non-null	object
184	ERW_4.6	2187	non-null	object
185	ERW_4.7	2187	non-null	object
186	ERW_4.8	2187	non-null	object
187	ERW_4.9	2183	non-null	object
188	ERW_4.10	2187	non-null	float64
189	ERW_4.11	2160	non-null	object
190	ERW_4.12	2187	non-null	object
191	ERW_4.13	2185	non-null	object
192	ERW_4.14	2187	non-null	object
193	ERW_4.15	2185	non-null	object
194	BIL_2.1	2187	non-null	float64
195	BIL_2.2	2187	non-null	float64
196	BIL_2.3	2187	non-null	float64
197	BIL_2.4	2187	non-null	object
198	BIL_3.1	2187	non-null	float64
199	BIL_3.2	2187	non-null	float64
200	BIL_3.3	2187	non-null	object
201	BIL_3.4	2187	non-null	object
202	BIL_3.5	2187	non-null	object
203	BIL_3.6	2187	non-null	object
204	BIL_3.7	2187	non-null	object
205	BIL_4.1	2187	non-null	float64
206	BIL_4.2	2187	non-null	float64
207	BIL_4.3	2187	non-null	object
208	BIL_4.4	2187	non-null	object
209	BIL_4.5	2187	non-null	float64
210	BIL_4.6	2187	non-null	float64
211	BIL_4.7	2187	non-null	float64
212	BIL_4.8	2187	non-null	object
213	BIL_4.9	2187	non-null	float64
214	BIL_4.10	2187	non-null	float64
215	BIL_5.1	2187	non-null	float64
216	BIL_5.2	2187	non-null	float64
217	BIL_5.3	2187	non-null	float64
218	BIL_5.4	2187	non-null	float64
219	BIL_5.5	2187	non-null	object
220	BIL_5.6	2187	non-null	float64
221	BIL_5.7	2187	non-null	object
222	BIL_5.8	2187	non-null	object

dtypes: category(8), float64(41), int64(8), object(166)

memory usage: 21.4+ MB

Tabelle 18: Erste zehn Zeilen Datensatz

	AGS	RS	Landkreis	RS	Landkreis	RS	Landkreis	Name	Reg.	HEV	DEM	1.1	BIL_4BIL_4BIL_5BIL_5BIL_5BIL_5BIL_5BIL_5	BIL_5.8
0	0	0	NaN	NaN	NaN	NaN	Deutsches	1800219695	553148402368909188339801792161032934985634710908970					
1	1	1	NaN	NaN	NaN	NaN	Schleswig-Holstein	280012800119	1889043297239946362801711295131110126590576225380					
2	1001000000	0	1	0	0	Flensburg	1825582258...	6200.0152607173022450322106330.0	3370.6210 /					
3	1001 1	0	1	NaN	NaN	Flensburg	1825582258...	6200.0152607173022450322106330.0	3370.6210 /					
4	1002000000	0	2	0	0	Kiel, Gemein-	280178235782	16290591402070665000820002329023660	10690193204100					
5	1002 1	0	2	NaN	NaN	Kiel, Stadt-	280178235782	16290591402070665000820002329023660	10690193204100					
6	1003000000	0	3	0	0	Lübeck	280178235782	1379036530183389527084470179902250	8490.131201800					
7	1003 1	0	3	NaN	NaN	Lübeck	280178235782	1379036530183389527084470179902250	8490.131201800					
8	1004000000	0	4	0	0	Neumünster	280178235782	4130.7650.6583019300347705920.660	2630.0230 /					
9	1004 1	0	4	NaN	NaN	Neumünster	280178235782	4130.7650.6583019300347705920.660	2630.0230 /					

Tabelle 19: Letzte zehn Zeilen Datensatz

	AGS	RS_L	RS_L	RS_N	RS_K	RS_V	RS_G	Name	Reg_Hier	NEW	DEM_1.1	BIL_4	BIL_4	BIL_5	BIL_5	BIL_5	BIL_5	BIL_5	BIL_5
12534	1,60773E+10	77	5009	26	Löbichau	Gemeinde	1041	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12535	1,60773E+10	77	5009	37	Nöbdenitz	Gemeinde	938	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12536	1,60773E+10	77	5009	41	Postersdorf	Gemeinde	446	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12537	1,60773E+10	77	5009	47	Thonhausen	Gemeinde	570	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12538	1,60773E+10	77	5009	49	Vollmarshausen	Gemeinde	328	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12539	1,60773E+10	77	5009	51	Wildenburg	Gemeinde	307	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12540	160775050	0	77	5050	NaN	Göbber	Gemeinde	543	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12541	1,60773E+10	77	5050	12	Göbber	Gemeinde	374	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12542	1,60773E+10	77	5050	17	Heyerdorf	Gemeinde	139	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12543	1,60773E+10	77	5050	39	Ponitz	Gemeinde	1600	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

5.3 Appendix df_analyse

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12544 entries, 0 to 12543
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Name                   12544 non-null  category
1   Reg_Hier               12544 non-null  category
2   Arbeitslosenquote     2187 non-null   float64
3   Migrationsquote       2187 non-null   float64
4   Christenquote         12544 non-null  float64
5   Männerquote           12544 non-null  float64
6   Akademikerquote       2187 non-null   float64
7   Beamtenquote          2187 non-null   float64
8   Singlequote           12544 non-null  float64
dtypes: category(2), float64(7)
memory usage: 1.0 MB
```

5.4 Appendix Dataframe bereinigt

Tabelle 20: Filter auf Ebene Gemeinde

	Name	Reg_Hier	Arbeitslosenquote	Migrationsquote	Christenquote	Männerquote	Akademikerquote	Beamtenquote	Singlequote
0	Flensburg, Stadt	Gemeinde	6.657547	15.957447	56.027377	49.276666	13.355639	8.378114	62.147147
1	Kiel, Landes-haupt-stadt	Gemeinde	7.539341	18.900021	48.656386	48.139807	17.758138	7.578323	64.350968
2	Lübeck, Hanse-stadt	Gemeinde	7.167394	16.812500	56.723806	47.470103	13.992802	6.540654	59.120801
3	Neumünster, Stadt	Gemeinde	6.899185	16.924489	56.149594	48.816166	8.385235	6.015860	56.965139
4	Brunsbüttel, Stadt	Gemeinde	5.365854	13.682565	62.607137	49.579243	6.877828	4.123711	51.277856
...
11289	Greiz, Stadt	Gemeinde	6.813820	2.112338	25.183037	47.736997	13.745338	3.089598	53.023676
11295	Zeulenroda, Gemeinde	Gemeinde	5.662651	3.613666	28.863238	48.219960	12.058824	4.342273	51.721678