Cluster-Analyse

Inhalt

Zielsetzung

Diese Arbeit soll zu einem besseren Verständnis der Landschaft deutscher Softwareentwickler beitragen. Anhand des Datensatzes der „Stackoverflow developer survey 2022“ sollen aussagekräftige Cluster ermittelt werden. Die Cluster sollen anhand von Unterschieden in Arbeitserfahrung und Beschäftigungsart ermittelt werden.

Datensatz

Beschreibung

Bei dem Datensatz handelt es sich um die frei zugänglichen Ergebnisse der „Stack Overflow Developer Survey 2022“. Die Umfrage wurde vom 11. Mai 2022 bis zum 1. Juni 2022 durchgeführt und hauptsächlich über Kanäle im Besitz von Stack Overflow beworben. Es wurde ein Multiple-Choice Format gewählt, gegliedert in die 7 Themenbereiche: 1. Grundlegende Informationen, 2. Bildung, Arbeit und Karriere, 3. Technologie und Tech-Culture, 4. Nutzung von Stack Overflow, 5. Demographische Informationen, 6. Arbeitsumfeld und 7. Abschließende Fragen. Insgesamt nahmen 73 268 Menschen an der Umfrage teil, mit einem Median der Bearbeitungszeit von 15,1 Minute. [1]

Für die erste Analyse wurden die Daten nach Personen gefiltert, die als Wohnort Deutschland angaben. In der zweiten Analyse wurden aus dieser Gruppe nur jene ausgewählt, welche die Fragen des „TeamsBranch“ bearbeiteten.

Aufbereitung der Daten

Kategoriale Daten

Kategoriale Daten (z. B. bevorzugte Programmiersprache(n)) wurden mithilfe von „one-hot-encoding“ aufbereitet: Dabei wird jeder Person für jede Antwortmöglichkeit eine 1 (ausgewählt) oder 0 (nicht ausgewählt) zugeordnet. Somit lassen sich auch Fragen verarbeiten, bei denen mehrere Antworten möglich waren. Das Nichtbearbeiten einer Frage (nan-Werte) wird als eigene Antwortmöglichkeit behandelt.

Ordinale kategoriale Daten

Trifft bspw. auf Fragen nach Zustimmungswerten zu. Dem niedrigsten Wert („stimme überhaupt nicht zu“) wird 0 zugeordnet, dem höchsten Wert („stimme vollkommen zu“) 1. Die verschiedenen Abstufungen werden in gleichen Intervallen zwischen 0 und 1 verteilt. Für Personen, die diese Fragen nicht bearbeiteten, wird der Median der übrigen Antworten eingesetzt. Hier könnte eine mögliche Quelle für Verzerrung des Datensatzes liegen, falls die entsprechenden Fragen begründet nicht beantwortet wurden.

Nominale Daten

Nominale Variablen (z. B. Alter) wurden auf das Intervall [0;1] normalisiert, sodass die entsprechenden Minima und Maxima bei 0, bzw. 1 liegen. Für fehlende Daten wurde wieder der Median des gesamten Datensatzes eingesetzt. Aufgrund vieler fehlender Angaben ist es hier möglicherweise sinnvoll, einen neuen Algorithmus auszuwählen, der nan-Werte handlen kann.

[1] Stack Overflow Developer Survey 2022

K-prototypes - Algorithmus

Beschreibung

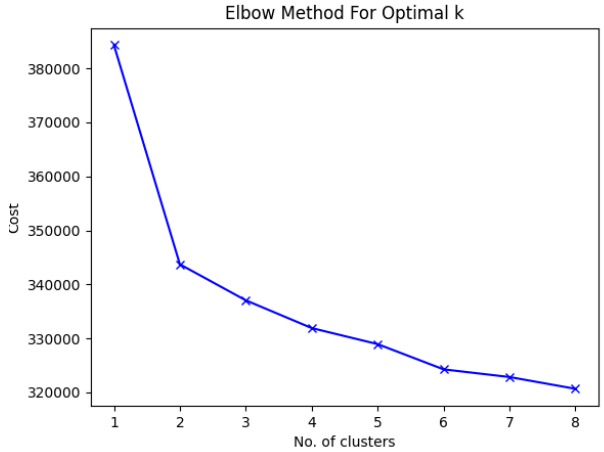
Als ersten Ansatz habe ich den k-prototypes Algorithmus gewählt. Hierbei handelt es sich um ein partitionierendes Clusterverfahren: die Anzahl k der Cluster wird im Vorhinein festgelegt. Hier habe ich zur Hilfe eine Kosten-Nutzung aufgestellt.

Der Algorithmus wählt anfangs k Clusterzentren mithilfe einer Dichtefunktion aus. Jeder Datenpunkt wird dem ihm nächsten Clusterzentrum zugeordnet. K-protoypes verwendet als Metrik sowohl die Euklidische Distanz für nominale Variablen, als auch die Menge an Unterschieden für kategoriale Variablen. Mithilfe einer Optimierungsfunktion wird anschließend die Lage der Clusterzentren so verändert, dass sich die durchschnittliche Entfernung jedes Datenpunktes zum jeweiligen Clusterzentrum minimiert. Dies geschieht so lange, bis sich die Zusammensetzung der Cluster nicht mehr ändert.

Auswahl der relevanten Kategorien

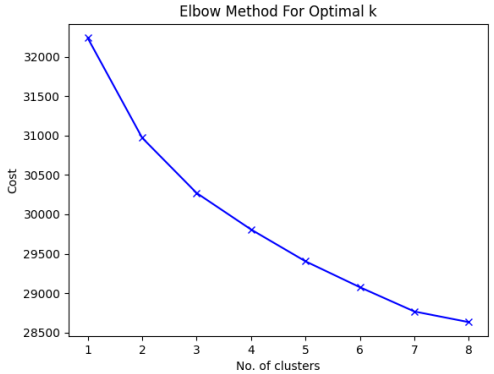
Bei dem ersten Clustering habe ich jede Frage gleichwertig in die Analyse miteinbezogen. Dabei stellte sich folgendes Problem heraus: die Stack-Overflow survey 2022 bestand aus einem ersten, generellen Teil und dem „TeamsBranch“, der Fragen speziell für professionelle Softwareentwickler enthält. Dieser Teil wurde allerdings nur von 60% der relevanten Gruppe bearbeitet. Dies erklärt den elbow-point in Abbildung 1.

Abbildung 1



Somit muss hier die Entscheidung getroffen werden, ob die Fragen des „Teams Branch“ von der Betrachtung ausgeschlossen werden, oder ob nur die entsprechende Untergruppe betrachtet wird. Ich habe mich vorerst dazu entschlossen, die Fragen des „TeamsBranch“ zu streichen, um die untersuchte Gruppe möglichst groß zu halten. Nach diesem Schritt erwies sich die elbow-method als annähernd linear (Abb. 2).

Abbildung 2



Dies legt nahe, dass sich mit erhöhter Clusterzahl die Komplexität nur linear reduzieren lässt, sich also keine statistischen Erkenntnisse gewinnen lassen. Deshalb habe ich mich dazu entschieden, mithilfe einer Korrelationsmatrix redundante Kategorien aus dem Clustering zu entfernen.

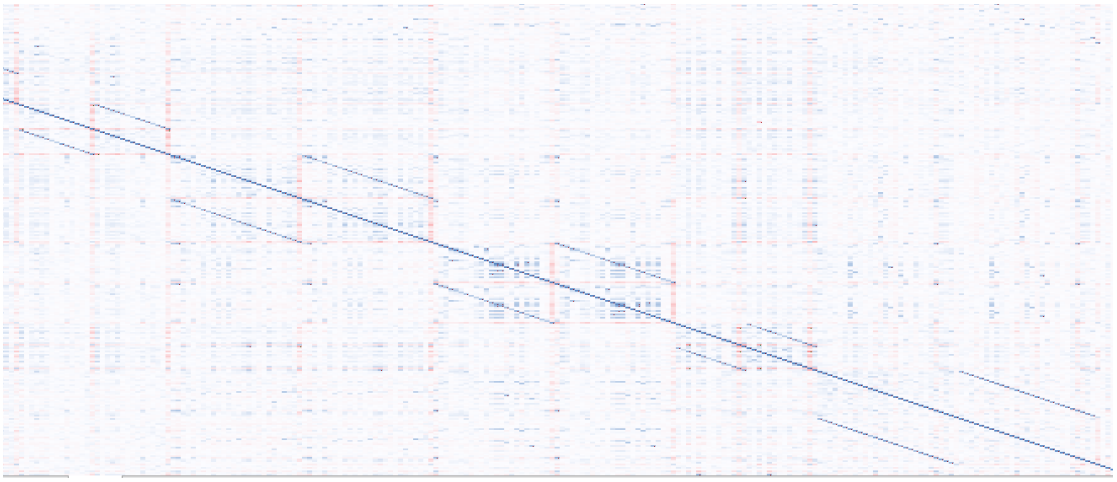


Abbildung 3

Die parallelen zur Hauptdiagonalen zeigen die jeweils hohe Korrelation zwischen „Software have worked with“ und „Software want to work with“ – um die Menge der Kategorien zu reduzieren, habe ich also alle Spalten mit „Software want to work with“ entfernt.

Fazit

Auch nach weiteren Vereinfachungen und Modifikationen erweist sich der k-prototypes – Algorithmus nicht als ausreichend. Die Diskrepanz zwischen den Gruppen bleibt zu gering, um statistisch relevante Aussagen treffen zu können.

Agglomerativ-hierarchischer Algorithmus (ohne TeamsBranch)

Beschreibung

Ich verwende den agglomerativen Algorithmus der scikit-learn Bibliothek in Python. Hierbei handelt es sich um einen bottom-up Ansatz: Zum Start des Programms wird jeder Datenpunkt als eigenes Cluster definiert. Diese werden fortlaufend vergrößert und gemerged, bis nur noch ein Cluster besteht oder eine vorher festgelegte Anzahl k an Clustern erreicht wird. Der agglomerative Algorithmus verwendet als Metrik die Euklidische Distanz. Deshalb ist es notwendig, die Daten zu normieren. In den Kategorien „Developer Type“ und „Education Level“ beträgt die maximale Distanz jeweils 2. Grund dafür ist, dass für beide Fragen nur eine Antwort möglich war. Im one-hot encoding könnte das so aussehen: *Person A: EdLevel_Bachelor (0); EdLevel_Master (1). Person B: EdLevel_Bachelor (1); EdLevel_Master(0).*

In der Kategorie „Developer Type“ waren mehrere Antworten möglich. Hier konnte ich eine maximale Distanz von 24 Einheiten messen. Entsprechend habe ich die Werte des one-hot encodings für „Developer Type“ von 1 (für True) auf 1/12 (für True) reduziert. So beträgt die maximale Distanz auch in dieser Kategorie 2.

Auswahl der Kategorien

Für diesen Ansatz werden nur die Kategorien „Developer Type“, „Main Branch“ und „Education Level“ verwendet. In der Auswertung werden alle deutschen Teilnehmer*innen berücksichtigt. Die Kategorie „Work Experience“ wird vorerst nicht berücksichtigt, da sie im „TeamsBranch“ der Umfrage stand, welcher nur von 49% der deutschen Teilnehmenden bearbeitet wurde.

Anzahl der Cluster

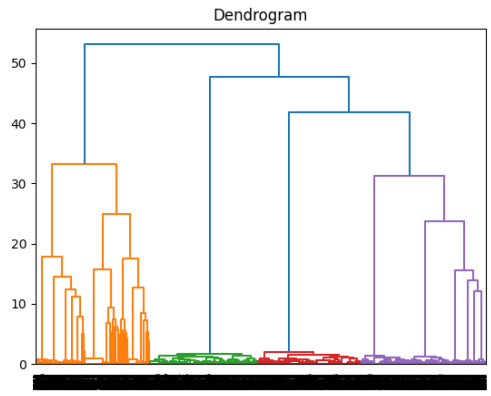


Abbildung 4

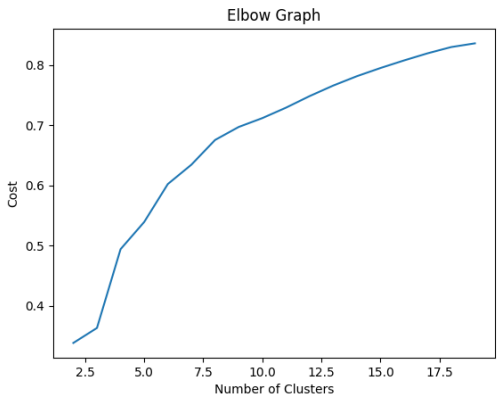


Abbildung 5

Das Dendogramm legt eine Einteilung in vier Cluster nahe. Dabei fällt auf, dass es sich um zwei besonders homogene Gruppen handelt (in der Mitte) und zwei vergleichsweise heterogene Gruppen. Auch der leichte Knick am elbow Graph (Abb.5) weist darauf hin, dass diese Anzahl der Cluster angemessen ist.

Beschreibung der Gruppen

Tabelle: cluster_results.xlsx

Im Folgenden werde ich die 3 entstandenen Gruppen charakterisieren. Neben den für das Clustering relevanten Kategorien (unterstrichen) beziehe ich weitere Daten in die Beschreibung mit ein. Die bevorzugten Programmiersprachen sind bspw. ein guter Indikator dafür, welchen Tätigkeiten die

Personengruppen nachgehen. Die Gehälter wurden von den Erstellern der Umfrage in Dollar umgerechnet und liegen somit höher als der originale Wert in Euro.

A: Ca. 28 Jahre *(min)* mit 10,5 Jahren Programmiererfahrung *(min)*. Unterdurchschnittliches Einkommen *(min)*. Das Programmieren wird nur zum Teil in der Arbeit angewendet, ist ein Hobby, oder wird gerade erlernt. Gemischte Bildungsschichten. Tätigkeitsbereiche weit gestreut, von 45% allerdings keine Angabe. Grund dafür könnte sein, dass diese Personen (noch) nicht in der Arbeitswelt tätig sind.

B: Ca. 31 Jahre mit 14 Jahren Programmiererfahrung. Unterdurchschnittliches Gehalt. Professionelle Software developer. Schulabschluss oder unvollendetes Studium (und 10% promoviert). Hohe Nutzung verschiedener Programmiersprachen, sowie hoher Einsatz über verschiedene Tätigkeitsbereiche hinweg.

C: Ca. 36 Jahre *(max)* mit 17 Jahren Programmiererfarung *(max)*. Überdurchschnittliches Gehalt *(max)*. Professionelle software developer mit Master-Abschluss. Im Vergleich zu **B** höhere Spezialisierung, mit Fokus auf back-end.

D: Ca 31 Jahre mit 12 Jahren Programmiererfahrung. Durchschnittliches Gehalt. Professionelle Software developer mit Bachelor Abschluss. Lassen sich bei Grad der Spezialisierung zwischen **B** und **C** einordnen.

	A	**B**	**C**	**D**	**Ø**
Größe (Anteil)	1364 (25,3 %)	1493 (27, 7 %)	1301 (24,1 %)	1237 (22,9 %)	1348,8
Alter	28,4	31,2	35,6	30,8	31,5
Programmier erfahrung (Jahre)	10,5	14,1	17	12,4	13,5
Einkommen ($)	76 843	85 701	118 246	97 532	94 022
Software development	Als Teil der Arbeit: 42,7% Hobby: 30,2% Lernend: 19%	Professionell: 100%	Professionell: 100%	Professionell: 100%	Professionell: 75% Teil der Arbeit: 10,8% Hobby: 7,6%
Bildung	Schulabschluss: 35,3 % Master: 20,2% Bachelor: 14,3%	Schulabschluss: 39,5% Uni ohne Abschluss: 33,9% Promoviert: 9,1%	Master: 100%	Bachelor: 100%	Master: 29,2% Bachelor: 26,5% Schulabschluss: 19,9%
Tätigkeit	Keine Angabe: 45% Forschung: 11,7% Schüler/Student: 10,9%	Full-stack dev.: 57,3 % Back-end dev.: 47% Front-end dev.: 30%	Back-end dev.: 46% Full-stack dev.: 43,2% Applicationdev.: 21,8%	Full-stack dev.: 50,1% Back-end dev.: 45,9% Front-end dev.: 27,9%	Full-stack dev.: 40,3% Back-end dev.: 36,6% Front-end dev.: 21,2%
Sprachen	Python:	JavaScript:	JavaScript:	JavaScript:	JavaScript:

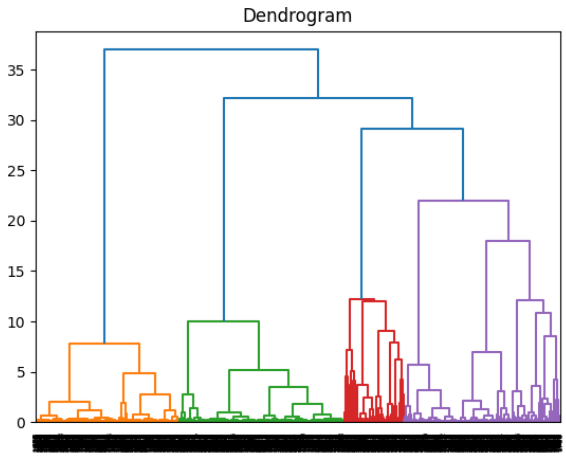
	61,6% JavaScript: 54% HTML/CSS: 50,5%	69% HTML/CSS: 62,6% SQL: 56,1%	52,4% Python: 51% SQL: 43,9%	64,8% HTML/CSS: 52,6% SQL: 47,3%	60,2% HTML/CSS: 52,6% Python: 49,6%
TeamsBranch	22,4%	52,3%	64%	58,4%	49%

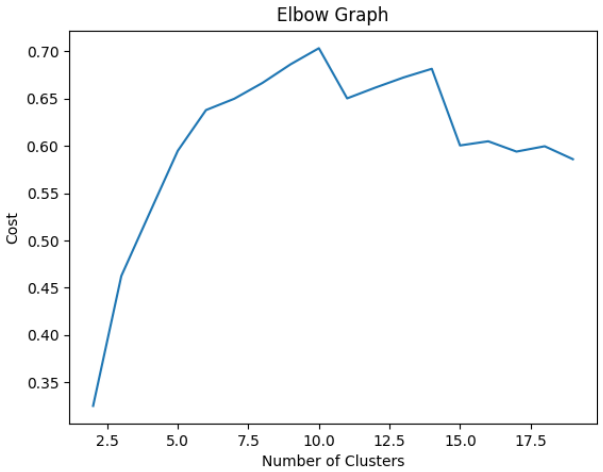
Agglomerativ-hierarchischer Algorithmus (mit TeamsBranch)

Auswahl der Kategorien

Zu den Kategorien der vorherigen Analyse konnte zusätzlich „Work Experience“ miteinbezogen werden. Auch diese Kategorie wird im Clustering auf einen Wert zwischen 0 und 2 skaliert.

Anzahl der Cluster





Während das Dendogramm eine Einteilung in 4 Cluster nahelegt, zeigt der Elbow-Graph, dass 2 Cluster tatsächlich die optimale Lösung darstellen. Ich habe mich für die größere Anzahl an Clustern entschieden. Die Zugehörigkeit der 4 Subgruppen zu den 2 größeren Gruppen lässt sich trotzdem leicht erkennen.

Charakterisierung der Gruppen

Tabelle: cluster_results_TeamsBranch.xlsx

A: Professionelle Softwareentwickler Anfang 30, mit ca. 10 Jahren Arbeitserfahrung. Die höchste Bildung ist der Schulabschluss oder ein nicht abgeschlossenes Studium. Unterdurchschnittliches Gehalt. Breit gefächertes Skillset und Einsatz vieler Programmiersprachen.

B: Personen, die nur als Teil ihrer Arbeit programmieren oder ehemals Programmierer waren. Mitte 30 mit ca. 11 Jahren Berufserfahrung. Bachelor-Abschluss oder höher. Gruppe mit dem niedrigsten Gehalt. Arbeit breit über verschiedene Einsatzgebiete gestreut (dabei 24% in Wissenschaft).

C: Professionelle Softwareentwickler Mitte 30 mit 11 Jahren Berufserfahrung. Älteste Gruppe. Masterabschluss. Höchstes Gehalt. Nutzung von Programmiersprachen selektiver, weniger breit gefächert. Fokus auf back-end.

D: Professionelle Softwareentwickler Anfang 30 (jüngste) mit 9 Jahren Berufserfahrung (wenigste). Bachelorabschluss. Durchschnittliches Gehalt. Bei Einsatzgebiet und Programmiersprachen ähnlich zu Gruppe A. Allerdings wieder höhere Spezialisierung (weniger Einsatzbereiche/Sprachen pro Person).

In der Tabelle sind die Werte zu sehen, auf die sich die Charakterisierungen stützen. Bei den unterstrichenen Kategorien handelt es sich um jene, die beim Clustering berücksichtigt wurden.

	A	B	C	D	Ø
Größe (Anteil gesamt)	787 (29,7 %)	300 (11,3 %)	833 (31,5 %)	723 (27,4 %)	660,8
Programmier-erfahrung (Jahre)	14,9	14,7	16,8	13,3	15
Arbeits-erfahrung (Jahre)	10,4	10,9	10,9	9	10,2
Einkommen ($)	87 987	81 145	122 561	105 209	102 818
Alter	32,1	34,2	35,3	31,8	33,3
Software development	Professionell: 100%	Als Teil der Arbeit: 79 % Ehemalig Professionell: 12 %	Professionell: 100%	Professionell: 100%	Professionell: 88,6 % Als Teil der Arbeit: 9%
Bildung	Schulabschluss: 35,3% Uni ohne Abschluss: 33,8% Promoviert: 10%	Master: 40,3% Promoviert: 17,7% Bachelor: 15,7%	Master: 100%	Bachelor: 100%	Master: 36,1% Bachelor: 29,1% Schulabschluss: 12,1%
Tätigkeit	Full-stack dev.: 62,6% Back-end dev.: 53,7% Front-end dev.: 33,7%	Forschung: 23,7% Systemadmin.: 20,7% Full-stack dev. : 18,7%	Back-end dev.: 46,6% Full-stack dev. : 44,9% Applicationsdev.: 21%	Full-stack dev.: 53,4% Back-end dev.: 51,2% Front-end dev.: 30,6%	Full-stack dev.: 49,5% Back-end dev.: 46,4% Front-end dev.: 25,8%
Sprachen	JavaScript: 69,9% HTML/CSS: 63,3% SQL: 60,2%	Python: 60% JavaScript: 49,7% HTML/CSS: 46%	JavaScript: 51,7% Python: 50,9% SQL: 44,9%	JavaScript: 68,6% HTML/CSS: 56% SQL: 49,5%	JavaScript: 61,5% HTML/CSS: 53,1% SQL: 50,7%

Fazit

Für den vorliegenden Datensatz zeigte sich der agglomerativ-hierarchische Algorithmus als geeigneter, die Personen in Gruppen mit statistischer Aussagekraft zu gliedern. Folgende allgemeine Erkenntnisse lassen sich aus der Analyse gewinnen:

1. Software developer mit Master-Abschluss verdienen im Durchschnitt am Meisten. Sie zeichnen sich außerdem durch eine hohe Spezialisierung und eine Fokussierung auf back-end aus.

2. Personen, die nur als Nebenaspekt ihres Berufs programmieren, verdienen am wenigsten. Dies gilt trotz hoher universitärer Abschlüsse. Die Sprache Python ist in dieser Gruppe stark verbreitet.
3. Je niedriger der Bildungsabschluss, desto niedriger liegt im Durchschnitt das Gehalt. Gleichzeitig geht mit niedrigem Bildungsabschluss eine hohe Diversifizierung und ein breit gefächertes Skillset einher. Sprachen für front-end und Datenverarbeitung sind hier weiter verbrietet.